Investment Incentives in Truthful Approximation Mechanisms*

Mohammad Akbarpour† Scott Duke Kominers‡
Kevin Michael Li§ Shengwu Li¶ Paul Milgrom‖

First posted: February 25, 2020
This version: May 16, 2022

Abstract

We study investment incentives in truthful mechanisms that allocate resources using approximation algorithms instead of exact optimization. In such mechanisms, the price a bidder pays to acquire resources is generally not equal to the change in other bidders' welfare—and these externalities skew investment incentives. Some allocation algorithms are arbitrarily close to efficient, but create such perverse investment incentives that their worst-case welfare guarantees fall to zero when bidders can invest before participating in the mechanism. We show that an algorithm's guarantees in the allocation and investment problems coincide if and only if that algorithm's "confirming" negative externalities are sufficiently small. Algorithms that exclude confirming negative externalities entirely (XCONE algorithms) thus have the same worst-case performance for the allocation and investment problems.

Keywords: Combinatorial optimization, Knapsack problem, Investment, Auctions, Approximation, Algorithms

JEL classification: D44, D47, D82

*We thank Moshe Babaioff, Peter Cramton, Michal Feldman, Matthew Gentzkow, Paul Goldsmith-Pinkham, Yannai Gonczarowski, Nima Haghpanah, Andy Haupt, John William Hatfield, Nicole Immorlica, Emir Kamenica, Zi Yang Kang, Eric Maskin, Ellen Muir, Rad Niazadeh, Noam Nisan, Amin Saberi, Mitchell Watt, numerous seminar audiences, the editor (Bart Lipman), and several referees for helpful comments. We thank Broadsheet Cafe for inspiration and coffee. Akbarpour and Kominers gratefully acknowledge the support of the Washington Center for Equitable Growth. Additionally, Kominers gratefully acknowledges the support the National Science Foundation (grant SES-1459912) and both the Ng Fund and the Mathematics in Economics Research Fund of the Harvard Center of Mathematical Sciences and Applications. Milgrom gratefully acknowledges support from the National Science Foundation (grant SES-1947514). All errors remain our own.

†Stanford University. Email: mohamwad@stanford.edu

‡Harvard University and a16z Crypto. Email: kominers@fas.harvard.edu

§Stanford University. Email: kli317@stanford.edu

¶Harvard University. Email: shengwu_li@fas.harvard.edu

‖Stanford University and Auctionomics. Email: milgrom@stanford.edu

1 Introduction

Many real-world allocation problems are too complex for exact optimization. For example, it is computationally difficult—even under full information—to optimally pack indivisible cargo for transport (Dantzig, 1957; Karp, 1972), coordinate electricity generation and transmission (Lavaei and Low, 2011; Bienstock and Verma, 2019), assign radio spectrum broadcast rights subject to legally-mandated interference constraints (Leyton-Brown et al., 2017), or find value-maximizing allocations in combinatorial auctions (Sandholm, 2002; Lehmann et al., 2006).

Computational difficulty has inspired research to identify fast approximation algorithms for the aforementioned allocation problems, and many others. The resulting algorithms, which take the participants' values for the resources as fixed, solve *short-run* problems, omitting *long-run* considerations about market participants incentives to make investments (or disinvestments) that change their values.

A bidder's investment incentives are shaped by the pricing rules associated to a given allocation mechanism. For example, suppose a bidder can pay a cost to raise its value for some outcome before participating in a mechanism. When facing any VCG mechanism, the extra profit that a bidder earns from an investment is always equal to the change in social welfare. For such a mechanism, any investment is privately profitable for a bidder if and only if it increases social welfare (Rogerson, 1992).

In this paper, we examine the investment incentives created by truthful mechanisms for computationally hard allocation problems. In our model, an *allocation* is a profile of outcomes—one for each bidder—chosen from an exogenous set of feasible allocations. Each bidder has quasi-linear utility with a private value vector specifying its payoff for each potential outcome, and the possible value vectors comprise a product of closed intervals. An *allocation algorithm* is a computational procedure that takes as input the value profile and the set of feasible allocations and then outputs an allocation. Each algorithm induces an allocation rule, that is, a function from inputs to outputs. Practical algorithms must run quickly, but our results do not depend on running time, so we use "algorithm" to refer both to the computational procedure and to the function that it induces. By a standard result, allocation algorithms can be converted into truthful mechanisms (with appropriate payments) if and only if the algorithm is weakly monotone.

When a bidder invests under a truthful mechanism, that bidder changes its reported value. We define the externality from that change to be the increase (positive, negative or zero) in the sum of other bidder's values for the resulting allocation, plus the increase in the price the bidder pays. For example, if the bidder's change in value reduces the welfare of other bidders but also increases the bidder's payment to the auctioneer by a larger absolute amount, we count that as a positive externality, because it increases the total welfare of the other bidders plus the auctioneer.

VCG mechanisms have zero externalities and, in that respect, are essentially unique. To be precise, let us say that an algorithm x is a *quasi-optimizer* if there is some subset of feasible allocations $\hat{A} \subseteq A$ such that for all value profiles v, the welfare of allocation $x(v, A)$ is equal to that of the best allocation from $\hat{A}$.[1] We prove that a truthful mechanism has no externalities if and only if its underlying allocation algorithm is a quasi-optimizer. Most standard algorithms for computationally difficult problems are not quasi-optimizers, so they induce non-zero externalities, with the result that some profitable investments lead to reduced welfare.

Since, as we show, any two truthful mechanisms that use the same underlying allocation rule result in identical externalities—and since our focus will be on allocation algorithms—we call these *algorithmic externalities*.

Algorithm researchers most often measure an algorithm's closeness to optimality in terms of a worst-case *performance ratio*, which is the minimum (over all input instances) of the algorithm's welfare divided by the optimal welfare. It is tempting to guess that if an algorithm has a high performance ratio in the short-run allocation problem, then it should also have a high performance ratio if we add an investment stage. But this guess can fail badly, as we illustrate with the classic *knapsack problem* (Dantzig, 1957).

An instance of the knapsack problem is described by a list of indivisible items, each having a positive size and a value, and a knapsack capacity. The problem is to select ("pack") a set of items to maximize the total value, subject to the sum of the item sizes not exceeding the capacity. For simplicity, suppose that each individual item's size is less than the knapsack's capacity. Finding the optimal solution to the knapsack problem is NP-hard.

Suppose that each item is associated with a different bidder and that the item sizes are publicly observed, but the value of being packed is the bidder's private information. An algorithm for the knapsack problem is *monotone* if raising any packed bidder's value leaves that bidder still being packed. Any monotone algorithm can be paired with a payment rule to create a truthful mechanism. One such payment rule—the *threshold rule*—charges zero to each unpacked bidder and charges each packed bidder its *threshold value*, which is the infimum of the values that would result in the bidder being packed.

The *greedy algorithm* for the knapsack problem arranges the items in decreasing order of value-to-size, packs items one-by-one in that order, and stops as soon as it encounters some

[1] A quasi-optimizer x can differ from the optimizer only on non-generic value profiles v for which $x(v, A)$ and the best allocation in $\hat{A}$ have the same welfare.

bidder	A	B	C
value	ε	1	.5
size	.4	.6	.5

Table 1: A knapsack instance, with capacity 1. Assume $0 < \varepsilon < .2$.

item that is too large to be included along with the items already packed. The *smart greedy algorithm* compares the value of this greedy packing with the highest value of any single item and takes the better of these two feasible solutions. By a textbook argument, for every knapsack instance, the value of the smart greedy solution is at least half the maximum, and this worst-case bound is tight—the algorithm's *performance ratio* is 0.5.[2]

Consider the following *satisficing algorithm* for the knapsack problem: If the most valuable item is worth at least 99% of the sum of all values, then pack that item and stop. Otherwise, solve the maximization exactly. This algorithm is monotone, so its corresponding threshold auction is truthful. By construction, the satisficing algorithm's performance ratio is .99.

When one of the bidders can make a costly investment before participating in the mechanism, a different picture emerges. The smart greedy algorithm's performance ratio remains unchanged at .5, while the satisficing algorithm's performance ratio falls all the way from .99 to 0.

To see why the satisficing algorithm performs so badly when investments are included, consider the knapsack instance specified by Table 1. The satisficing algorithm packs bidders A and B, for a total welfare of $1 + \varepsilon$. Since Bidder A is packed for any positive ε, A's threshold price is 0. Suppose that Bidder A can invest at a cost of 200, to raise its value to $200 + 2\varepsilon$. This investment is profitable for Bidder A, and it causes the satisficing algorithm to pack just A, for net welfare of $200+2\varepsilon-200 = 2\varepsilon$. But the social optimum is to invest and pack both A and B, for net welfare $1 + 2\varepsilon$. The performance ratio with private investment is therefore no more than $\frac{2\varepsilon}{1+2\varepsilon}$, which can be arbitrarily close to 0.

By contrast, the smart greedy algorithm's worst-case guarantee is robust to the possibility of investment. In the Table 1 example, the smart greedy algorithm packs just B. Bidder A's threshold price is .4, so it is not profitable to make the investment—and net welfare is 1, compared to the optimum of $1 + 2\varepsilon$. Our main theorem implies that in truthful mechanisms based on the smart greedy algorithm, the net welfare when the first bidder chooses investments to maximize profit is always at least .5 of the maximum net welfare. This guarantee

[2] The argument is as follows. Suppose we relax the integer constraints; the solution to the resulting linear program (LP) is an upper bound for the maximum. The LP solution consists of the greedy solution plus a fraction of the next item on the list. The smart greedy solution is weakly better than the greedy solution and also weakly better than the next item on the list, so it achieves at least 0.5 of the LP solution.

holds for any investment technology for the first bidder and any values for the other bidders.

What distinguishes these two algorithms that accounts for such different worst-case performance? Both are approximation algorithms that sometimes fail to maximize welfare. Both pack either the most valuable single item or a selection of items, depending on an inequality condition. The important difference is that, for the satisficing algorithm, bidder A's investment both *confirms* A's original outcome and causes a negative externality, reducing the welfare of bidder B by 1; yet, as we show, no such externalities arise for the smart greedy algorithm. For any algorithm without such *confirming negative externalities*, the investment performance guarantee is the same as the allocation performance guarantee.[3]

For our general treatment, we assume that investments may be made with uncertainty about all the future inputs to the algorithm: the values resulting from the investment, the values of the other bidders, and the set of feasible allocations. Each of the inputs is allowed to be a general function of some finite state space S with any probability distribution, provided that the realization in every state is an allowable instance of the deterministic problem. The investor selects an investment to maximize its own expected payoff, net of the price it pays and its investment costs. We compare the expected welfare of the truthful mechanism with self-interested investment to the maximum expected welfare in the same problem, under optimal investment and optimal allocation. The algorithm is a *β-approximation for the investment problem* with state space S if for all probabilities of the states, the expected performance of the mechanism is at least β times the optimum.

We prove that uncertainty can be excluded in assessing investment performance: x is a β-approximation for investment for all finite state spaces S *if and only if* the same property holds for all singleton state spaces (that is, for $|S| = 1$).

Focusing on investment without uncertainty, our results utilize two new concepts—"confirming changes" and their associated "confirming negative externalities." Suppose that at some value profile (v_n, v_{-n}), the algorithm allocates outcome o to bidder n. A change in bidder n's values from v_n to $\tilde{v}_n$ is *confirming* if for any other outcome $\hat{o}$, we have $\tilde{v}_n^o - v_n^o \geq \tilde{v}_n^{\hat{o}} - v_n^{\hat{o}}$, that is, n's value for the original outcome increases at least as much as its value for any other outcome. If, given such a confirming change to n's report, the algorithm's allocation changes in a way that results in a negative externality, we call that a *confirming negative externality.*

Our first main result establishes a necessary and sufficient condition for the investment

[3]For problems like the knapsack problem with just two outcomes for each bidder, confirming externalities coincide with bossy externalities. An agent's changed report is *bossy* if it changes others' outcomes but does not change the bidder's own outcome. With more than two outcomes for each bidder, some confirming externalities are not bossy, and the confirming externality concept is the one we need to state and prove our theorems.

and allocative guarantees to coincide. That condition is a bound on magnitude of confirming negative externalities. Suppose we start at value profile (v_n, v_{-n}) and make a confirming change to $\tilde{v}_n$; our condition requires that any resulting negative externality must not exceed the slack in the allocative guarantee β^* at the original value profile. This bound can be hard to assess, however, because the slack depends on the optimal welfare, which is hard to compute or characterize for many problems of interest. The second result is a corollary of the first, that is easier to check because it requires no knowledge about the optimal welfare: if an algorithm excludes confirming negative externalities (XCONE), then its investment and allocative guarantees coincide.

To provide intuition for our results, we specialize this introductory discussion to the case of packing problems, in which bidder n faces only two outcomes—winning (being "packed") or losing—when its price for being packed is p. The worst-case investment scenarios involve a bidder who invests c and makes a zero profit; indeed, any reduction in c that further increases the bidder's profit only increases the performance ratio, so it is never the worst case.

Consider a bidder n who would not be packed without investment but by investing at cost $c > 0$ can increase its value to $p + c$, resulting in a net profit of zero. The intuitive argument hinges on decomposing this investment into two parts. Suppose that the bidder first has the option of investing at zero cost to raise its value just to the threshold price p. This is a zero-profit investment, since even if the bidder is packed, it must pay its full value p for that. Because the investment cost is zero, this investment problem has the same performance as an allocation problem without investment, so its performance ratio is some $\beta \geq \beta^*$. In the actual problem, bidder n can invest $c > 0$ to increase its value above the threshold to $p + c$. This option does not increase the investor's net payoff and does not increase welfare in the welfare-maximization problem. Thus, this investment, which confirms n's packing affects the performance ratio if and only if it results in an externality. If the externality is positive, then the performance ratio must rise, resulting in $\beta > \beta^*$. For the performance ratio to fall, the confirming externality must be negative—and to bring performance down to some $\beta < \beta^*$, the negative externality must be sufficiently large. In particular, an XCONE algorithm, which excludes confirming negative externalities entirely, cannot lead to the investment performance falling below β^*.

Some standard approximation algorithms are XCONE, including the Greedy algorithm, the Smart Greedy algorithm, and the clock auction algorithm used for the 2017 Federal Communication Commission's broadcast incentive auction (Leyton-Brown et al. (2017), Milgrom and Segal (2020)). Meanwhile, we establish that the well-known *minimum spanning tree algorithm* for the Steiner tree problem is not XCONE.

We also show that XCONE is closely related to non-bossiness. An algorithm is non-bossy if a change in one bidder's report that does not affect that bidder's outcome also does not change any other bidder's outcome. For allocation problems with two outcomes, such as the knapsack problem, every non-bossy algorithm is XCONE. Even some bossy algorithms (such as Smart Greedy) are XCONE; we show that any algorithm that takes the best output from a family of non-bossy algorithms is XCONE.[4]

What if more than one bidder can make investments? Even with a VCG mechanism, the investment game can have multiple equilibria—and in some equilibria, bidders can fail to coordinate on the efficient investment profile. However, VCG mechanisms satisfy a different efficiency property: For any belief that one bidder has about the other bidders' investment choices, that bidder's selfish investment is interim efficient, maximizing social welfare net of that bidder's investment cost conditional on whatever the investor may know. Our results extend this property to approximation algorithms: For any truthful mechanism based on an XCONE algorithm with allocative guarantee β^*, each bidder's selfish investment choice achieves welfare (net of that bidder's investment costs) that is within β^* of the interim optimum.

1.1 Related work

Economists have studied *ex ante* investment in mechanism design at least since the work of Rogerson (1992), who demonstrated that Vickrey mechanisms induce efficient investment. Bergemann and Välimäki (2002) extended this finding in a setting with uncertainty, in which bidders invest in information before participating in an auction. Relatedly, Arozamena and Cantillon (2004), studied pre-market investment in procurement auctions, showing that while second-price auctions induce efficient investment, first-price auctions do not. Hatfield et al. (2014, 2019) extended these findings to characterize a relationship between the degree to which a mechanism fails to be truthful and/or efficient and the degree to which it fails to induce efficient investment. While like us, Hatfield et al. (2014, 2019) dealt with the connection between (near-)efficiency at the allocation stage and (near-)efficiency at the investment stage, they use additive error bounds, rather than the multiplicative worst-case bounds that are standard for the analysis of computationally hard problems. Gershkov et al. (2020) studied the construction of revenue-maximizing mechanisms with *ex ante* investment. Tomoeda (2019) studied full implementation of exactly-efficient social choice rules with investment.

Our paper is also not the first work to study investment incentives in an NP-hard allocation setting. Milgrom (2017) introduced a "knapsack problem with investment" in which the

[4] This result extends to problems with more than two outcomes under a tie-breaking condition.

items to be packed are owned by individuals, and owners may invest to make their items either more valuable or smaller (and thus easier to fit into the knapsack). In the present paper, we reformulate the investment question in terms of worst-case guarantees and broaden the formulation to study incentive-compatible mechanisms for a wide class of resource allocation problems.

Lipsey and Lancaster (1956) explained that in economic systems that are not fully optimized, investments that violate optimality conditions can sometimes improve welfare by offsetting other distortions of the system. Our question is related to that of Lipsey and Lancaster (1956), but leads to a different analysis. We isolate *bossy* negative externalities as the only externalities that can degrade an allocation algorithm's long-run performance guarantee relative to its short-run guarantee. Other externalities associated with failures of optimization cannot have that effect.

By studying the investment problem in near-optimal mechanisms, our paper is naturally connected to a large literature, primarily in computer science, that considers computational complexity in mechanism design, and explores properties of approximately optimal mechanisms. Among these works are those of Nisan and Ronen (2007) and Lehmann et al. (2002). Nisan and Ronen (2007) showed that in settings where identifying the optimal allocation is an NP-hard problem, VCG-based mechanisms with nearly-optimal allocation algorithms are generically non-truthful, while Lehmann et al. (2002) introduced a truthful mechanism for the knapsack problem in which the allocation is determined by a greedy algorithm. In addition, Hartline and Lucier (2015) developed a method for converting a (non-optimal) algorithm for optimization into a Bayesian incentive compatible mechanism with weakly higher social welfare or revenue; Dughmi et al. (2017) generalized this result to multidimensional types. For a more comprehensive review of results on approximation in mechanism design, see Hartline (2016).

There is also a large literature on greedy algorithms of the type we study here, which sort bidders based on some intuitive criteria and choose them for packing in an irreversible way; see Pardalos et al. (2013) for a review. Lehmann et al. (2002) studied the problem of constructing truthful mechanisms from greedy algorithms; similarly, Bikhchandani et al. (2011) and Milgrom and Segal (2020) proposed clock auction implementations of greedy allocation algorithms.

Finally, our concept of an XCONE algorithm is closely related to the definition of a "bitonic" algorithm, introduced by Mu'Alem and Nisan (2008) to construct truthful mechanisms in combinatorial auctions. Bitonicity is defined for binary outcomes; with the restriction to binary outcomes, every XCONE algorithm is bitonic, but not vice versa.

2 The model

2.1 Approximation algorithms

Consider a set of **bidders** N and a set of **outcomes** O, both finite. For instance, in the knapsack problem, the set of outcomes is $\{\text{packed}, \text{unpacked}\}$. The value of bidder n for outcome o is $v_n^o \in \mathbb{R}_{\geq 0}$. We write $v_n \equiv (v_n^o)_{o\in O}$ to denote n's **values**, and we write $v \equiv (v_n)_{n\in N}$ to denote a full **value profile**. An **allocation** $a \in O^N$ assigns one outcome to each bidder; a_n denotes the outcome of bidder n.

An **allocation instance** (N, O, v, A) consists of a set of bidders N, a set of outcomes O, a value profile v, and a set of **feasible allocations** $A \subseteq O^N$. To simplify notation, we often write instances as a pair (v, A), leaving N and O implicit.

The standard approach in computer science is to assess an algorithm's worst-case performance over a domain of instances. Hence, we define an **allocation problem** Ω to be a collection of instances. We require that the value profiles in Ω have a product structure. That is, let V_n^o be a closed interval of $\mathbb{R}_{\geq 0}$ capturing the possible values that bidder n might have for outcome o. We define $V_n \equiv \prod_{o\in O} V_n^o$ and require that $\{v : (N, O, v, A) \in \Omega\} = \prod_{n\in N} V_n$. In some settings, one outcome $o \in O$ is an **outside option** known to be valued at 0; we capture this with $V_n^o = \{0\}$.

An **algorithm** x is a function that selects a feasible allocation for each instance $(v, A) \in \Omega$, that is, $x(v, A) \in A$.[5] We denote the outcome assigned to bidder n under x by $x_n(v, A)$. We abuse notation and identify outcomes o with binary vectors of length $|O|$, with one element equal to 1 and all others equal to 0, which allows us to write the **welfare** of algorithm x on instance (v, A) as

$$W_x(v, A) \equiv \sum_n \left[v_n \cdot x_n(v, A)\right].$$

The **optimal welfare** at instance (v, A) is

$$W^*(v, A) \equiv \max_{a\in A} \left\{ \sum_n \left[v_n \cdot a_n\right] \right\}.$$

Given some $\beta \in [0, 1]$, algorithm x is a **β-approximation for allocation** if for all

[5] In complexity theory, we often are not given the feasible allocations A directly, but instead only a description that implies which allocations are feasible. For instance, a description could specify the bidders' sizes and the capacity of the knapsack. In principle, algorithms for the knapsack problem could output different allocations for two instances with different item sizes but the same feasible allocations. Our formulation ignores this description-dependence, but we could easily accommodate it by specifying a function from descriptions to feasible allocations, and defining an instance as consisting of a value profile v and a description d; none of our results would materially change with this adjustment.

$(v, A) \in \Omega$, we have that $W_x(v, A) \geq \beta W^*(v, A)$. We refer to the largest such β as the algorithm's **allocative guarantee**.

2.2 Truthful mechanisms

Suppose that the bidder's values are private information, so that the algorithm cannot directly input each bidder n's value v_n but must instead rely on each bidder's *reported* value $\hat{v}_n$. To elicit these reports, we use a **mechanism** (x, p), which is a pair consisting of an algorithm x and a payment rule p that maps any reported instance $(\hat{v}, A)$ into an allocation $x(\hat{v}, A) \in A$ and a profile of payments $p(\hat{v}, A) \in \mathbb{R}^N$. We adopt the sign convention that payments are made by the participants and to the auctioneer. A mechanism is **truthful** if for all instances $(v, A) \in \Omega$ and all $\hat{v}_n \in V_n$, we have that

$$v_n \cdot x_n(v, A) - p_n(v, A) \geq v_n \cdot x_n(\hat{v}_n, v_{-n}, A) - p_n(\hat{v}_n, v_{-n}, A).$$

When can an algorithm be paired with a pricing rule to produce a truthful mechanism? Algorithm x is **weakly monotone** (**W-Mon**) if for any two instances (v, A) and $(\tilde{v}_n, v_{-n}, A)$, we have

$$[\tilde{v}_n - v_n] \cdot [x_n(\tilde{v}_n, v_{-n}, A) - x_n(v, A)] \geq 0. \tag{1}$$

For packing problems, we have $\{O\} = \{1, 0\}$, a value for being "packed" $v_n^1 \geq 0$, and an outside option with $v_n^0 = 0$. In this special case, (1) reduces to the requirement that, under x, if n is packed at (v_n, v_{-n}, A) and $\tilde{v}_n^1 \geq v_n^1$, then n is packed at $(\tilde{v}_n, v_{-n}, A)$.

A necessary condition for the existence of a pricing rule p such that (x, p) is truthful is that the algorithm is weakly monotone, and since V_n is convex, this is also sufficient, as we state below.[6]

Lemma 2.1 (Lavi et al. (2003); Saks and Yu (2005))**.** *An algorithm x is weakly monotone if and only if there exists a payment rule p such that (x, p) is truthful.*

Pricing rules in truthful mechanisms can be defined in terms of threshold prices, one for each outcome. The least value for bidder n to achieve outcome o is denoted by

$$\hat{\tau}_n^o(v_{-n}, A, x) = \inf_{v_n \in V_n} \{v_n^o : x_n(v_n, v_{-n}, A) = o\}$$

and the **threshold price** is

$$\tau_n^o(v_{-n}, A, x) \equiv \min\left\{\hat{\tau}_n^o(v_{-n}, A, x), \sup V_n^o\right\}.$$

[6] Bikhchandani et al. (2006) provided other domain assumptions such that W-Mon is sufficient.

Our results hold trivially if $\tau_n^o(v_{-n}, A, x) = \infty$.[7] To focus on the non-trivial case, we assume that $\tau_n^o(v_{-n}, A, x) < \infty$; by V_n^o closed, it follows that $\tau_n^o(v_{-n}, A, x) \in V_n^o$. We denote the **threshold vector** by $\tau_n(v_{-n}, A, x) \equiv (\tau_n^o(v_{-n}, A, x))_{o\in O}$. The set of possible values V_n has a product structure, so we have $\tau_n(v_{-n}, A, x) \in V_n$.

V_n is path-connected, so a standard argument by the envelope theorem yields the following lemma (Milgrom and Segal, 2002).

Lemma 2.2. *If (x, p) is a truthful mechanism, then for each n, there exists a real-valued function $f_n(v_{-n}, A)$ such that*

$$p_n(v, A) = \tau_n(v_{-n}, A, x) \cdot x_n(v, A) + f_n(v_{-n}, A).$$

Lemma 2.2 states that in a truthful mechanism, each bidder pays the threshold price to achieve its assigned outcome plus a strategically irrelevant term that does not depend on the bidder's own report. We say that (x, p) is a **threshold mechanism** if $f_n(v_{-n}, A) \equiv 0$ everywhere. Truthfulness of (x, p) implies that x assigns each bidder an outcome that maximizes its value minus threshold price, that is, for all n and (v, A), we have

$$[v_n - \tau_n(v_{-n}, A, x)] \cdot x_n(v, A) = \max_{o\in O} \{[v_n - \tau_n(v_{-n}, A, x)] \cdot o\}.$$

2.3 Algorithmic externalities

Given mechanism (x, p) and instance (v, A), the **externality** of changing n's value from v_n to $\tilde{v}_n$ is

$$\mathcal{E}_{x,p}(\tilde{v}_n, (v, A)) \equiv \underbrace{p_n(\tilde{v}_n, v_{-n}, A) - p_n(v, A)}_{\text{change in } n\text{'s payment}} + \underbrace{\sum_{m\neq n} v_m \cdot [x_m(\tilde{v}_n, v_{-n}, A) - x_m(v, A)]}_{\text{effect on others' welfare}}. \quad (2)$$

Expression (2) is the portion of n's effect on other participants' welfare that is not fully reflected by n's price.[8] Equivalently, if we treat the auctioneer as the residual claimant to any surplus or deficit of the mechanism, then (2) is the change in the sum of the payoffs of other participants, including the auctioneer.

Lemma 2.2 implies that any two truthful mechanisms that use the same allocation algorithm x have the same externalities. Consequently, we henceforth suppress the dependence

[7] Observe that if $\tau_n^o(v_{-n}, A, x) = \infty$, then $\hat{\tau}_n^o(v_{-n}, A, x) = \infty$ and $\sup V_n^o = \infty$, *i.e.* bidder n is never allocated outcome o and can have arbitrarily large values for o, which in turn implies that x is not a β-approximation for allocation for any $\beta > 0$.

[8] In the some parts of the economics and mechanism design literatures, the word "externality" is used to refer just to the second term, but our definition is faithful to the traditional Pigouvian concept of externality.

of $\mathcal{E}_{x,p}$ on p, writing $\mathcal{E}_x$ and calling this an **algorithmic externality**. VCG mechanisms have zero externalities, so it follows that x has zero algorithmic externalities if x is exactly maximizing.

We characterize the zero-externality algorithms. Let us say that x is a **quasi-optimizer** if for each set of feasible allocations A, there exists $B \subseteq A$ such that for all v we have

$$W_x(v, A) = \max_{a \in B} \left\{ \sum_n [v_n \cdot a_n] \right\}.$$

Theorem 2.1. *If algorithm x is weakly monotone, then x has no algorithmic externalities ($\mathcal{E}_x \equiv 0$) if and only if x is a quasi-optimizer.*

Most practical approximation algorithms are not quasi-optimizers, so it is (essentially) only VCG mechanisms that have no algorithmic externalities. In mechanisms with algorithmic externalities, selfish investment decisions do not always maximize social welfare. Next we study the connection between algorithmic externalities and performance guarantees under investment.

2.4 Performance under investment

Given a truthful mechanism (x, p), we assess whether the mechanism's performance guarantee applies also to long-run problems in which a single bidder, denoted $\iota \in N$, can decide whether to invest and/or what investment to make. In our formulation, the bidder may be uncertain about what the situation will be when the mechanism is run, including potential uncertainty as to the values that would result from each of its possible investments, the values of the other bidders, and the feasible set that will apply. We compare the expected social welfare from the bidder's selfish investment choice and the given mechanism's allocation to the expected welfare from making the *ex ante* efficient investment and using the *ex post* efficient allocation.

We model the investor's uncertainty using a probability space with a finite number of states S. Each uncertain **investment** opportunity is a pair (ν_ι, c) consisting of a function $\nu_\iota : S \to V_\iota$ and a cost $c \in \mathbb{R}$, while other bidders' values and the feasible set are the function $\nu_{-\iota} : S \to V_{-\iota}$ and correspondence $\mathcal{A} : S \rightrightarrows A$. We do not restrict the correlations among these uncertain elements.

Formally, an **investment instance** $(N, O, S, g, \iota, I, \nu_{-\iota}, \mathcal{A})$ consists of:

1. Sets of bidders N and outcomes O.

2. A finite set of states S and a probability distribution $g \in \Delta S$.

3. An **investor** $\iota \in N$.

4. A finite set of investments I. To represent the status quo, we require that this set includes at least one pair (ν_ι, c) with $c = 0$.

5. A function from states to the other bidders' values, $\nu_{-\iota} : S \to V_{-\iota}$.

6. A correspondence from states to feasible allocations, $\mathcal{A} : S \rightrightarrows O^N$.

We require that each state $s \in S$ and each investment $(\nu_\iota, c) \in I$ together result in an instance of the original allocation problem, i.e., that $(N, O, \nu(s), \mathcal{A}(s)) \in \Omega$. To simplify our notation, we write each investment instance in the form $\overline{\omega} = (g, \iota, I, \nu_{-\iota}, \mathcal{A})$, suppressing N, O, and S. We use a line above variables to distinguish functions or variables related to an investment problem, so $\overline{\omega}$ indicates that this refers to an investment instance.

Suppose that the investor participates in some truthful mechanism (x, p). After an investment is chosen and the state is realized, the investor can do no better than to report the resulting value truthfully to the mechanism. Hence, its best response investment choice at instance $\overline{\omega} = (g, I, \nu_{-\iota}, \mathcal{A})$ is

$$\mathrm{BR}(x, p, \overline{\omega}) \equiv \operatorname*{argmax}_{(\nu_\iota, c) \in I} \left\{ \left(\sum_{s \in S} g(s) \left[\nu_\iota(s) \cdot x_\iota(\nu_\iota(s), \nu_{-\iota}(s), \mathcal{A}(s)) - p_\iota(\nu_\iota(s), \nu_{-\iota}(s), \mathcal{A}(s)) \right] \right) - c \right\}.$$

By Lemma 2.2, the price $p_\iota(v, A)$ paid by the investor consists of a term entirely pinned down by the algorithm x, plus a term that does not depend on its own report. Thus, for any two truthful mechanisms that use the same algorithm, (x, p) and (x, p'), the bidder has the same privately optimal investments—$\mathrm{BR}(x, p, \overline{\omega}) = \mathrm{BR}(x, p', \overline{\omega})$—so we henceforth suppress the payment rule argument p from $\mathrm{BR}(\cdot)$.

The **welfare** of algorithm x at investment instance $\overline{\omega} = (g, I, \nu_{-\iota}, \mathcal{A})$ is

$$\overline{W}_x(\overline{\omega}) \equiv \min_{(\nu_\iota, c) \in \mathrm{BR}(x, \overline{\omega})} \left\{ \left(\sum_{s \in S} g(s) W_x(\nu_\iota(s), \nu_{-\iota}(s), \mathcal{A}(s)) \right) - c \right\}.$$

We benchmark performance relative to the net welfare delivered by *ex ante* efficient investment and *ex post* efficient allocations. That is, the **optimal welfare** at investment instance $\overline{\omega} = (g, I, \nu_{-\iota}, A)$ is

$$\overline{W}^*(\overline{\omega}) \equiv \max_{(\nu_\iota, c) \in I} \left\{ \left(\sum_{s \in S} g(s) W^*(\nu_\iota(s), \nu_{-\iota}(s), \mathcal{A}(s)) \right) - c \right\}.$$

Given some $\beta \in [0, 1]$, algorithm x is a **β-approximation for investment** if for every investment instance $\overline{\omega}$, we have that $\overline{W}_x(\overline{\omega}) \geq \beta \overline{W}^*(\overline{\omega})$. Notably, since we are quantifying over $\iota \in N$ and I, this requires the performance guarantee β to hold regardless of which bidder is the investor and which investments are available.[9]

Adding investment opportunities weakly reduces the algorithm's performance guarantee.

Proposition 2.1. *If x is a β-approximation for investment, then x is a β-approximation for allocation.*

Proof. Any instance of the allocation problem (v, A) is equivalent to the instance of the investment problem $(g, I, \nu_{-\iota}, \mathcal{A})$ with the singleton investment technology $I = \{(\nu_\iota, 0)\}$, $\nu_\iota \equiv v_\iota$, $\nu_{-\iota} \equiv v_{-\iota}$, and $\mathcal{A} \equiv A$; the result then follows. □

The converse of Proposition 2.1 does not hold in general—investment opportunities may strictly reduce the algorithm's performance guarantee. But when is an algorithm's performance guarantee robust to the introduction of investment opportunities? We now determine the answer.

First, we simplify the problem by observing that, for our purposes, it is without loss of generality to focus on the case without uncertainty. That is, a **certain investment instance** $\overline{\omega}$ is an investment instance with just one state, so $|S| = 1$, and we abuse notation by writing such an instance as $(I, v_{-\iota}, A)$. An algorithm is a β**-approximation for certain investment** if $\overline{W}_x(\overline{\omega}) \geq \beta \overline{W}^*(\overline{\omega})$ for any certain investment instance $\overline{\omega}$.

The next theorem states that an algorithm's performance guarantee with investment is the same as its performance guarantee with certain investment.

Theorem 2.2. *For any weakly monotone algorithm x and any $\beta \in [0, 1]$, x is a β-approximation for investment if and only if x is a β-approximation for certain investment.*

The intuition for Theorem 2.2 is as follows: Suppose we start from some investment instance with uncertainty. We can construct a related investment instance that replaces each investment with one that yields the same values in every state but instead has a *state-dependent* cost that makes the realized profit in each state equal to the original *ex ante* expected profit. This change leaves the expected profit and costs from investing unchanged, so selfish investment in the original instance yield the same expected welfare as selfish investment in the new instance. If the algorithm is a β-approximation for certain investment, then in the new instance, in every state the selfish investment achieves at least a fraction β of the welfare from the *ex post* efficient investment and the *ex post* efficient allocation. In turn, this is an upper bound for the expected welfare from the *ex ante* efficient investment

[9] Our results extend naturally if some bidders are known in advance to be unable to make investments. In that case, our necessary conditions weaken to pertain only to those bidders who can make investments.

and the *ex post* efficient allocation in the new instance, which is the same as in the original instance. It follows that x is a β-approximation for investment.

We next derive a necessary and sufficient condition for an algorithm x to be a β-approximation for certain investment.

If mechanism (x, p) is truthful, then bidder n's assigned outcome $x_n(v, A)$ maximizes its value net of threshold price $v_n^o - \tau_n^o(v_{-n}, A, x)$. Thus, we define the **relative utility** of bidder n at allocation instance (v, A) as

$$u_n(v, A, x) \equiv \max_{o \in O} \left\{ v_n^o - \tau_n^o(v_{-n}, A, x) \right\}.$$

By construction, we have $u_n(v, A, x) \geq 0$.

Suppose the zero-investment allocation instance is (v, A) and the allocative guarantee is β. This guarantee compares the algorithm's welfare $W_x(v, A)$ to the optimum welfare at the same instance $W^*(v, A)$, requiring $W_x(v, A) \geq \beta W^*(v, A)$. We now define a related condition that compares $W_x(v, A)$ to the optimal welfare when one bidder's value is set to the threshold vector, plus that bidder's relative utility.

Definition 2.1. For some $\beta \in [0, 1]$, algorithm x is β**-pivotal** if for any allocation instance (v, A) and any bidder n, we have

$$W_x(v, A) \geq \beta \underbrace{W^*(\tau_n(v_{-n}, A, x), v_{-n}, A)}_{\text{optimal welfare at } n\text{'s threshold vector}} + u_n(v, A, x). \tag{3}$$

The next theorem states that the β-pivotality condition is necessary and sufficient for an algorithm to attain performance guarantee β under certain investment.

Theorem 2.3. *For any weakly monotone algorithm x and any $\beta \in [0, 1]$, x is β-pivotal if and only if x is a β-approximation for certain investment.*

Next we relate β-pivotality to algorithmic externalities. We can simplify the problem by focusing on value changes in certain directions. We say that changing from v_n to $\tilde{v}_n$ **confirms** outcome $\tilde{o}$ if

$$[\tilde{v}_n - v_n] \cdot [\tilde{o} - o] \geq 0 \text{ for all outcomes } o. \tag{4}$$

Intuitively, (4) means that changing n's value from v_n to $\tilde{v}_n$ raises n's value for $\tilde{o}$ at least as much as n's value for any other outcome—equivalently, n's marginal gain from switching from o to $\tilde{o}$ does not fall. The system of inequalities (4) defines a convex cone with vertex at v_n. If x is weakly monotone, then any change from v_n to $\tilde{v}_n$ that confirms $x_n(v, A)$ implies that

$$[\tilde{v}_n - v_n] \cdot [x_n(\tilde{v}_n, v_{-n}, A) - x_n(v, A)] = 0. \tag{5}$$

For any truthful (x, p), type v_n cannot profitably imitate $\tilde{v}_n$ and *vice versa*, so

$$4v_n \cdot [x_n(\tilde{v}_n, v_{-n}, A) - x_n(v, A)] \leq p_n(\tilde{v}_n, v_{-n}, A) - p_n(v, A) \leq \tilde{v}_n \cdot [x_n(\tilde{v}_n, v_{-n}, A) - x_n(v, A)]. \tag{6}$$

From (5) and (6), it follows that

$$p_n(\tilde{v}_n, v_{-n}, A) - p_n(v, A) = v_n \cdot [x_n(\tilde{v}_n, v_{-n}, A) - x_n(v, A)]\,; \tag{7}$$

that is, the bidder with value v_n is indifferent between reporting v_n and reporting $\tilde{v}_n$ when facing (v_{-n}, A).

The externalities from confirming changes reduce to a simple expression. In particular, they are equal to the difference between the welfare yielded by the new allocation at the old values and the welfare yielded by the old allocation at the old values.

Proposition 2.2. *For any weakly monotone x, any instance (v, A), and any change from v_n to $\tilde{v}_n$ that confirms $x_n(v, A)$, we have*

$$\mathcal{E}_x(\tilde{v}_n, (v, A)) = \sum_m [v_m \cdot [x_m(\tilde{v}_n, v_{-n}, A) - x_m(v, A)]]\,. \tag{8}$$

Proof. Substituting (7) into (2) yields (8). □

Definition 2.2. For some $\beta \in [0, 1]$, algorithm x has **β-bounded confirming externalities** if given any instance (v, A) and any change from v_n to $\tilde{v}_n$ that confirms $x_n(v, A)$, we have

$$\mathcal{E}_x(\tilde{v}_n, (v, A)) \geq \beta W^*(v, A) - W_x(v, A). \tag{9}$$

The inequality (9) requires the algorithmic externality of the confirming change to exceed the lower bound, which is the negative of the slack in the allocative guarantee at instance (v, A).

We can now relate β-pivotality to the lower bound on confirming externalities that we have just defined.

Theorem 2.4. *For any weakly monotone algorithm x and any $\beta \in [0, 1]$, x is β-pivotal if and only if x has β-bounded confirming externalities.*

We summarize the preceding results in the following corollary:

Corollary 2.1. For any weakly monotone algorithm x and any $\beta \in [0,1]$, the following statements are equivalent:

1. x is a β-approximation for investment.
2. x is a β-approximation for certain investment.
3. x is β-pivotal.
4. x has β-bounded confirming externalities.

We highlight two characteristics of this description of performance guarantees under investment. First, the β-pivotality inequality (3) compares performance across two different instances (v, A) and $(\tau_n(v_{-n}, A, x), v_{-n}, A)$, whereas an allocative performance guarantee compares algorithms holding the instance fixed. This highlights how good allocative guarantees are not sufficient for good performance under investment. Second, expressions (3) and (9) require that we assess the *optimal* welfare at some instance. But our main interest is in those problems for which optimal allocations are hard to compute. Thus, even though Theorems 2.3 and 2.4 yield necessary and sufficient conditions for an algorithm to attain some guarantee β, direct application of these results may sometimes be intractable.

We thus introduce a tractable sufficient condition that implies that any guarantee for the allocation problem also holds for every investment instance.

Definition 2.3. Algorithm x **excludes confirming negative externalities** (**"is XCONE"**) if given any instance (v, A) and any change from v_n to $\tilde{v}_n$ that confirms $x_n(v, A)$, we have

$$\mathcal{E}_x(\tilde{v}_n, (v, A)) \geq 0.$$

Theorem 2.5. *For any weakly monotone algorithm x and any $\beta \in [0,1]$, if x is XCONE and a β-approximation for allocation, then x is a β-approximation for investment.*

Proof. From the assumption that x is XCONE, we know that the left-hand side of (9) is at least 0. Meanwhile, the fact that x is a β-approximation for allocation implies that the right-hand side of (9) is no more than 0. Thus, we see that x has β-bounded confirming externalities; by Corollary 2.1, it follows that x is a β-approximation for investment. □

2.4.1 Relation to non-bossiness

Our XCONE condition is related to the standard mechanism design concept of non-bossiness. Algorithm x is **non-bossy** if $x_n(\tilde{v}_n, v_{-n}, A) = x_n(v, A)$ implies that $x(\tilde{v}_n, v_{-n}, A) = x(v, A)$, that is, if changing n's value does not change n's outcome it must not change others' outcomes, either. Algorithm x is **consistent** if (5) implies that $x_n(\tilde{v}_n, v_{-n}, A) = x_n(v, A)$; this requires that for any given v_{-n} and associated threshold prices, when bidder n is indifferent among outcomes, the algorithm breaks those ties in a consistent way.[10]

Proposition 2.3. *If x is weakly monotone, consistent, and non-bossy, then x is XCONE.*

Proof. Because x is weakly monotone (4) implies (5). Meanwhile, (5) implies that $x(\tilde{v}_n, v_{-n}, A) = x(v, A)$ because x consistent and non-bossy. Thus, we have

$$0 = \sum_m v_m \cdot [x_m(\tilde{v}_n, v_{-n}, A) - x_m(v, A)] = \mathcal{E}_x(\tilde{v}_n, (v, A)),$$

where the final equality follows by Proposition 2.2. □

2.4.2 Combinations of XCONE algorithms

A standard technique for computationally hard problems is to run several candidate algorithms and select the best of their solutions; this can yield a better allocative guarantee than each individual algorithm. However, the resulting algorithm may be bossy, even if the candidate algorithms are non-bossy. By contrast, if the candidate algorithms are XCONE, then the resulting algorithm is XCONE.

Proposition 2.4. *Let X be a collection of weakly monotone XCONE algorithms. If y is a weakly monotone algorithm that at each instance $(v, A) \in \Omega$ outputs a welfare-maximizing allocation from the collection $\{x(v, A)\}_{x \in X}$, then y is XCONE.*[11]

Proof. Consider any instance (v, A) and any $\tilde{v}_n$ that confirms $x(v, A)$, Let $x \in X$ be such that $y(v, A) = x(v, A)$. Because x is weakly monotone and XCONE, Proposition 2.2 implies that

$$0 \leq \mathcal{E}_x(\tilde{v}_n, (v, A)) = \sum_m v_m \cdot [x_m(\tilde{v}_n, v_{-n}, A) - x_m(v, A)]\,;$$

[10] This requirement holds trivially for packing problems, since there is only one value v_n that is indifferent between being packed and being unpacked.

[11] Our necessary and sufficient condition, Definition 2.2, also has this property. That is, if we replace the supposition that every algorithm in X is XCONE with the supposition that every algorithm in X has β-bounded confirming externalities, then a parallel proof yields the conclusion that y has β-bounded confirming externalities.

hence,

$$\begin{aligned}\sum_m v_m \cdot y_m(v, A) &= \sum_m v_m \cdot x_m(v, A) \\ &\leq \sum_m v_m \cdot x_m(\tilde{v}_n, v_{-n}, A) \\ &\leq \sum_m v_m \cdot y_m(\tilde{v}_n, v_{-n}, A).\end{aligned}$$

Rearranging the preceding expression yields

$$0 \leq \sum_m v_m \cdot [y_m(\tilde{v}_n, v_{-n}, A) - y_m(v, A)] = \mathcal{E}_y(\tilde{v}_n, (v, A)),$$

where the equality follows from Proposition 2.2 because y is weakly monotone. □

Proposition 2.4 assumes that y is weakly monotone. Yet weak monotonicity of every algorithm in X does not imply weak monotonicity of y. One other advantage of XCONE algorithms is that when there are two outcomes, y inherits weak monotonicity from candidate XCONE algorithms.

Proposition 2.5. *Assume $|O| = 2$. Let X be a collection of weakly monotone XCONE algorithms. If y is an algorithm that at each instance $(v, A) \in \Omega$ outputs a welfare-maximizing allocation from the collection $\{x(v, A)\}_{x \in X}$, then y is weakly monotone.*

Proposition 2.5 does not generalize to $|O| > 2$. There exist pairs of candidate algorithms, both weakly monotone and XCONE, such that the resulting y is not weakly monotone—as the following example illustrates.

Example 2.1. Consider an allocation problem with two bidders and three outcomes, and suppose that $V_1 = [0, 4] \times [0, 4] \times \{0\}$ and $V_2 = \{(5, 0, 0)\}$. We suppose that algorithm x always allocates outcome 2 to bidder 1 and outcome 3 to bidder 2, while algorithm $\tilde{x}$ allocates outcome 1 to both bidders if $v_1 \geq 1$ and allocates outcome 3 to both bidders otherwise. Both x and $\tilde{x}$ are weakly monotone and XCONE. Let y be an algorithm that outputs a welfare-maximizing allocation from the set $\{x(v_1, v_2), \tilde{x}(v_1, v_2)\}$. Under algorithm y, bidder 1 gets outcome 2 when $v_1 = (0, 1, 0)$ and outcome 1 when $v_1 = (2, 4, 0)$, so y is not weakly monotone.

2.4.3 Allowing multiple investors

Suppose that each bidder n has a finite set of feasible investments I_n and, as before, an investment consists of a function $\nu_n : S \to V_n$ and a cost $c \in \mathbb{R}$. Suppose that all bidders simultaneously choose investments, knowing that in each state $s \in S$, the resulting allocation and payments will be $x(\nu(s), A(s))$ and $p(\nu(s), A(s))$, for truthful mechanism (x, p). The resulting investment game has a Nash equilibrium, possibly in mixed strategies.

Even for VCG mechanisms, not every Nash equilibrium of the investment game is efficient, as the following example illustrates.

Example 2.2. There is a knapsack with capacity 1, and three bidders, with sizes $q_1 = 1$, $q_2 = q_3 = 0.5$. There is only one state and so no uncertainty: $|S| = 1$. Bidder 1 has a *status quo* value 10 for being packed, that is, its technology is the singleton $I_1 = \{(10, 0)\}$. Bidders 2 and 3 have the technology $I_2 = I_3 = \{(0, 0), (9, 1)\}$. Total welfare is maximized if both bidders 2 and 3 choose the investment $(9, 1)$, which leads to both being packed. However, if only one of them invests, then it is optimal to pack just Bidder 1. In the VCG auction, there are two pure strategy Nash equilibrium investment profiles. In one Nash equilibrium, no bidder invests and Bidder 1 is packed, for net welfare 10. In the efficient Nash equilibrium, both Bidders 2 and 3 invest and both are packed, for net welfare 16.

Nevertheless, VCG mechanisms satisfy a different efficiency criterion: Conditional on any belief about the strategies of the other bidders, every best response for bidder n maximizes interim social welfare net of bidder n's investment costs.

Our results extend this observation to include approximate efficiency. Any belief about other bidders' investment choices can be represented by an expanded state space $S \times S'$ and functions $\tilde{\nu}_{-n} : S \times S' \to V_{-n}$. For each of bidder n's investments $(\nu, c) \in I_n$, we define a corresponding investment $(\tilde{\nu}_n, c)$ with $\tilde{\nu}_n(s, s') \equiv \nu_n(s)$, and similarly define $\tilde{\mathcal{A}}(s, s') \equiv \mathcal{A}(s)$. It follows from Theorem 2.3 and Theorem 2.2 that if x is β-pivotal, then any best response of bidder n to its belief about the other bidder's investments yields social welfare (net of n's investment costs) that is at least a fraction β of what would be achieved by the interim efficient investment for bidder n and the *ex post* efficient allocation.

3 Application: Knapsack algorithms

The **knapsack problem** is a special case of the allocation problem introduced in Section 2.1. There are two outcomes, $\{1, 0\}$, corresponding to being packed and being unpacked. Each bidder n has possible values $V_n^1 = [0, \infty)$ and $V_n^0 = \{0\}$. We abuse notation and use v_n to denote v_n^1, bidder n's value for being packed, since $v_n^0 \equiv 0$.

Each bidder has **size** $q_n \geq 0$, and the knapsack has **capacity** Q. Without loss of generality, suppose no bidder's size is more than Q. The set of feasible allocations is any subset of bidders $K \subseteq N$ such that $\sum_{n \in K} q_n \leq Q$. As before, let A denote the set of feasible allocations and let a be an element of A.

The knapsack problem is NP-Hard (Karp, 1972); there is no known polynomial-time algorithm that outputs optimal allocations (Cook, 2006; Fortnow, 2009). Dantzig (1957) suggested applying a **Greedy algorithm** to the knapsack problem. Formally:

Algorithm 1 (Greedy)**.** Sort bidders by the ratio of their values to their sizes so that

$$\frac{v_1}{q_1} \geq \frac{v_2}{q_2} \cdots \geq \frac{v_{|N|}}{q_{|N|}}. \tag{10}$$

Add bidders to the knapsack one by one in the sorted order, so long as the sum of the sizes does not exceed the knapsack's capacity. When encountering the first bidder that would violate the capacity constraint, stop.

Although the Greedy algorithm performs well on some instances, including ones for which all bidders are small in relation to the capacity of the knapsack, its worst-case performance guarantee is 0, as illustrated by the following example.

Example 3.1. Consider a knapsack with capacity 1 and two bidders. For some arbitrarily small $\epsilon > 0$, let $v_1 = \epsilon$, $q_1 = \frac{\epsilon}{2}$, $v_2 = 1$, and $q_2 = 1$. The Greedy algorithm picks bidder 1 and stops, whereas the optimal algorithm picks bidder 2. Thus, Greedy's performance is no better than ϵ of the optimum.

There is a standard modification of the Greedy algorithm that improves the worst-case guarantee for the knapsack problem. Let us define the **"smart greedy" algorithm** as follows.

Algorithm 2 (SmartGreedy)**.** Run the Greedy algorithm. Compare the Greedy algorithm's packing to the the most valuable individual bidder, and output whichever has higher welfare.

SmartGreedy's worst-case performance is much better than Greedy's:

Proposition 3.1. SmartGreedy *is a* $\frac{1}{2}$*-approximation for the Knapsack problem.*

Proof. For any instance ω, order the bidders by value/size as in (10). If Greedy packs all bidders, then trivially $W^*(\omega) = W_{\text{SmartGreedy}}(\omega)$. Otherwise, let k be the lowest index of a bidder not packed by Greedy and let K be the index of a bidder with maximum value. Optimal welfare $W^*(\omega)$ is no more than the best solution to the linear program in which we can pack fractional bidders, which—given that we have sorted the bidders in descending order of value-to-size—in turn is no more than $\sum_{n=1}^{k} v_n$. It follows that

$$\begin{aligned} W^*(\omega) &\le \sum_{n=1}^{k} v_n \\ &= W_{\text{Greedy}}(\omega) + v_k \\ &\le W_{\text{Greedy}}(\omega) + v_K \\ &\le 2\max\{W_{\text{Greedy}}(\omega), v_K\} \\ &= 2W_{\text{SmartGreedy}}(\omega). \end{aligned}$$ □

SmartGreedy is bossy, as our next example shows.

Example 3.2. Consider the knapsack instance with capacity 10 and 3 bidders, $v_1 = 2$, $v_2 = 1$, $v_3 = 8$, $q_1 = q_2 = 1$, and $q_3 = 9$. At this instance, SmartGreedy packs just bidder 3. If bidder 3 instead reports $v_3 = 10$, then SmartGreedy instead packs bidder 1 and bidder 3. Thus, SmartGreedy is bossy. However, note also that this is a confirming *positive* externality; raising the value of a packed bidder has strictly increased the welfare of other bidders.

Proposition 3.2. *For the knapsack problem, the* Greedy *algorithm and the* SmartGreedy *algorithm are both XCONE.*

Proof. Consider the bidders sorted by the Greedy algorithm as in (10), and suppose the Greedy algorithm packs bidders 1 through K. If we raise the value of a packed bidder, then the Greedy algorithm terminates at the same point, packing bidders 1 through K. If we lower the value of an unpacked bidder, then the Greedy algorithm terminates no earlier than before, packing at least bidders 1 through K. The only confirming externalities are positive ones; hence the Greedy algorithm is XCONE.

The SmartGreedy algorithm's output is equal to the welfare-maximizing selection from the outputs of $N + 1$ algorithms:

- the Greedy algorithm, and
- the algorithm that selects item n ($n = 1, ..., N$).

The Greedy algorithm is monotone and XCONE and so, trivially, are the N other algorithms. Thus, by Proposition 2.5, the SmartGreedy algorithm is monotone, and so by Proposition 2.4 it is XCONE. □

There can be negative externalities under the SmartGreedy algorithm. Suppose we have three bidders with sizes $(.5, .5, .6)$ and values $(1, 1, 0)$, and a knapsack with capacity 1. The SmartGreedy algorithm packs the first two bidders. Raising the third bidder's value to 2 raises its payment by 1.2 but reduces the welfare of the other bidders by 2. However, this value change is not confirming, so the resulting negative externality cannot undermine the SmartGreedy algorithm's worst-case performance guarantee of $\frac{1}{2}$. Conversely, Example 3.2 shows that there can be confirming externalities under the SmartGreedy algorithm, but because those confirming externalities are not negative, they cannot undermine the worst-case performance guarantee.

Some standard optimization algorithms are not XCONE. For example, in Appendix B, we show that well-known minimum spanning tree algorithm for the Steiner tree problem is not XCONE.

4 Discussion

Mechanism design analysis in economics has traditionally focused mostly on mechanisms that exactly optimize some objective like welfare, revenue, or consumer surplus, neglecting issues of computational hardness. For VCG mechanisms, computing truthful prices requires solving an additional optimization for every participant, adding to the computational burden. Yet exact optimization is tractable only for small problems or problems with special structure, and using approximate optimization to estimate truthful prices can lead to wildly inappropriate incentives (Milgrom and Segal, 2020).

Practical mechanisms without optimization can be created by utilizing the large corpus of fast approximation algorithms developed by computer scientists, but doing that raises new questions. Computer scientists' algorithms are nearly always designed for short-run problems in which some critical resources are held fixed. For example, approximation algorithms can be useful for scheduling landing slots at a crowded airport during bad weather, in which the flights and runways are given. What incentivizes investments? If inputs to approximation algorithms are elicited using a truthful mechanism, how might those prices affect the airlines' incentives to schedule flights or the airport operators incentives to design and build runways? In this paper, we asked three general questions about that:

1. *Can mechanisms based on approximation algorithms avoid distorting participants' investment incentives, as VCG mechanisms, based on optimization, do?*

2. *Can mechanisms based on approximation algorithms preserve their performance guarantees when an investment state is added, at least in the case of no uncertainty?*

3. *And how does uncertainty about investment returns, other bidders' values, and future feasible sets alter these conclusions?*

To frame the first question, we began by showing that that the externalities from any truthful mechanism depend only on the algorithm, and not on which prices are used to promote truthful reporting. For that reason, we call these "algorithmic externalities." Then, for the first question, we find a negative answer: unless the algorithm maximizes welfare on some possibly limited set of allocations, there are necessarily non-zero externalities that can cause privately profitable investments to reduce welfare or welfare-increasing investments to be privately unprofitable.

The analysis of performance without uncertainty hinges on a new category of externalities that we dub "confirming" externalities. These arise when a bidder changes its report in a way that raises the relative value of its outcome but results in an externality to others. We show that the worst-case performance guarantee is robust to investments if and only if its confirming negative externalities are not too large. That condition, however, can be hard to check, so we also offer a sufficient condition—XCONE—that is often easier to check. An XCONE algorithm is one that excludes confirming negative externalities, but may have confirming positive externalities. We show that, for some algorithms for the knapsack problem including Greedy and MGreedy, the XCONE condition can be checked and verified without much difficulty. However, the XCONE condition also fails for some algorithms with very good—even arbitrarily good—performance for the short-run allocation problem.

For the third question, it is undeniable that an algorithm's expected performance must depend on details about environmental uncertainty. For example, the expected returns to an airline's investment in airplanes or flights depends on the investor's expectations about traffic, weather, airport capacity, and its competitor's decisions, among other variables. Yet we show that, despite this dependence, the worst-case performance for an investment problem under uncertainty is the same as the worst case performance with no uncertainty.

More broadly, there is a long tradition in economics of studying the performance of a competitive equilibrium in which all decisions, short-run and long-run, are guided by optimization. An important way to generalize that conception is to introduce approximation algorithms instead of optimization, but that raises new considerations. Approximation can affect how participants understand mechanisms in practice, raise new opportunities for coordination or collusion, and influence post-auction resale markets. Given the close connection between weakly monotone algorithms and truthful mechanisms, it seems possible—and important—to analyze how these and other economic properties may arise from or correspond to properties of the underlying algorithms themselves.

References

Arozamena, L. and E. Cantillon (2004): "Investment incentives in procurement auctions," *Review of Economic Studies*, 71, 1–18.

Bergemann, D. and J. Välimäki (2002): "Information Acquisition and Efficient Mechanism Design," *Econometrica*, 70, 1007–1033.

Bienstock, D. and A. Verma (2019): "Strong NP-hardness of AC power flows feasibility," *Operations Research Letters*, 47, 494–501.

Bikhchandani, S., S. Chatterji, R. Lavi, A. Mu'alem, N. Nisan, and A. Sen (2006): "Weak Monotonicity Characterizes Deterministic Dominant-Strategy Implementation," *Econometrica*, 74, 1109–1132.

Bikhchandani, S., S. De Vries, J. Schummer, and R. V. Vohra (2011): "An ascending vickrey auction for selling bases of a matroid," *Operations Research*, 59, 400–413.

Cook, S. (2006): "The P versus NP problem," in *The Millennium Prize Problems*, ed. by J. A. Carlson, A. Jaffe, and A. Wiles, American Mathematical Society Providence, 87–104.

Dantzig, G. B. (1957): "Discrete-variable extremum problems," *Operations research*, 5, 266–288.

Dughmi, S., J. D. Hartline, R. Kleinberg, and R. Niazadeh (2017): "Bernoulli factories and black-box reductions in mechanism design," in *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*, 158–169.

Fortnow, L. (2009): "The status of the P versus NP problem," *Communications of the ACM*, 52, 78–86.

Gershkov, A., B. Moldovanu, P. Strack, and M. Zhang (2020): "A Theory of Auctions with Endogenous Valuations," *Journal of Political Economy.*

Hartline, J. D. (2016): "Mechanism design and approximation," *Book draft*, 122.

Hartline, J. D. and B. Lucier (2015): "Non-optimal mechanism design," *American Economic Review*, 105, 3102–24.

Hatfield, J. W., F. Kojima, and S. D. Kominers (2014): "Investment Incentives in Labor Market Matching," *American Economic Review Papers & Proceedings*, 104, 436–441.

——— (2019): "Strategy-Proofness, Investment Efficiency, and Marginal Returns: An Equivalence," Becker Friedman Institute Working Paper.

Karp, R. M. (1972): "Reducibility among combinatorial problems," in *Complexity of Computer Computations*, ed. by R. E. Miller and J. W. Thatcher, Springer, 85–103.

Lavaei, J. and S. H. Low (2011): "Zero duality gap in optimal power flow problem," *IEEE Transactions on Power Systems*, 27, 92–107.

Lavi, R., A. Mu'Alem, and N. Nisan (2003): "Towards a characterization of truthful combinatorial auctions," in *Proceedings of the 44th Annual IEEE Symposium on Foundations of Computer Science*, 574–583.

Lehmann, D., R. Müller, and T. Sandholm (2006): "The Winner Determination Problem," in *Combinatorial Auctions*, ed. by P. Cramton, Y. Shoham, and R. Steinberg, MIT Press, 297–318.

Lehmann, D., L. I. Oćallaghan, and Y. Shoham (2002): "Truth revelation in approximately efficient combinatorial auctions," *Journal of the ACM*, 49, 577–602.

Leyton-Brown, K., P. Milgrom, and I. Segal (2017): "Economics and computer science of a radio spectrum reallocation," *Proceedings of the National Academy of Sciences*, 114, 7202–7209.

Lipsey, R. and K. Lancaster (1956): "The General Theory of Second Best," *Review of Economic Studies*, 24, 11–32.

Milgrom, P. (2017): *Discovering Prices*, Columbia University Press.

Milgrom, P. and I. Segal (2002): "Envelope theorems for arbitrary choice sets," *Econometrica*, 70, 583–601.

——— (2020): "Clock auctions and radio spectrum reallocation," *Journal of Political Economy*, 128, 1–31.

Mu'Alem, A. and N. Nisan (2008): "Truthful approximation mechanisms for restricted combinatorial auctions," *Games and Economic Behavior*, 64, 612–631.

Nisan, N. and A. Ronen (2007): "Computationally feasible VCG mechanisms," *Journal of Artificial Intelligence Research*, 29, 19–47.

Pardalos, P. M., D.-Z. Du, and R. L. Graham (2013): *Handbook of Combinatorial Optimization*, Springer.

Rogerson, W. P. (1992): "Contractual solutions to the hold-up problem," *Review of Economic Studies*, 59, 777–793.

Saks, M. and L. Yu (2005): "Weak monotonicity suffices for truthfulness on convex domains," in *Proceedings of the 6th ACM Conference on Electronic Commerce*, 286–293.

Sandholm, T. (2002): "Algorithm for optimal winner determination in combinatorial auctions," *Artificial Intelligence*, 135, 1–54.

Tomoeda, K. (2019): "Efficient investments in the implementation problem," *Journal of Economic Theory*, 182, 247–278.

Vazirani, V. V. (2013): *Approximation Algorithms*, Springer Science & Business Media.

Vickrey, W. (1961): "Counterspeculation, auctions, and competitive sealed tenders," *The Journal of Finance*, 16, 8–37.

A Proofs omitted from the main text

Proof of Theorem 2.1

If x is a quasi-optimizer, then it acts like a VCG mechanism on some set of feasible allocations and therefore has zero externalities.

To show the other direction, we need some definitions to identify the subset of feasible allocations over which the algorithm x is a quasi-optimizer. Two allocations a, a' are *equivalent* denoted, $a \simeq a'$, if for any value profile, both allocations yield equal total welfare.[12] The *modified domain* denoted, $S \subseteq V$, is the set of value profiles v where for any $a, a' \in A$

$$W(v, a) = W(v, a') \implies a \simeq a'$$

and a *modified range*, denoted $R \subseteq A$, that is the range of x restricted to the modified domain. We will show that x is a quasi-optimizer over the modified range.

The proof relies on the following two lemmas. The first characterizes the welfare function for an algorithm that has no externalities. The second shows that the modified domain is quite dense.

[12] Since V has a product structure, it is only possible for different allocations to be equivalent if there are multiple bidders that have multiple outcomes that take singleton values.

Lemma A.1. *If algorithm x has zero externalities, then W_x is nondecreasing and 1-Lipschitz.*

Proof. Since x has zero externalities, we can rewrite 2 as

$$W_x(v_n, v_{-n}, A) - u_n(v_n, v_{-n}, A, x) = W_x(\tilde{v}_n, v_{-n}, A) - u_n(\tilde{v}_n, v_{-n}, A, x)$$

This means $W_x(\cdot, v_{-n}, A)$ is a constant plus $u_n(\cdot, v_{-n}, A)$. It is clear from the definition that $u_n(\cdot, v_{-n}, A)$ is nondecreasing and 1-Lipschitz therefore W_x is nondecreasing and 1-Lipschitz as well. □

Lemma A.2. *For any values $v \in V$, $s \in S$ and $\epsilon > 0$, we can construct v' such that $||v' - v|| < \epsilon$ and $v^* \in S$ for any v^* such that $v^*_{n,o} \in \{s_{n,o}, v'_{n,o}\}$.*

Proof. Call a value v' good if $v^* \in S$ for any v^* such that $v^*_{n,o} \in \{s_{n,o}, v'_{n,o}\}$. Start with $v' = s$. We can change the value $v'_{n,o}$ to be within $\frac{\epsilon}{N \cdot O}$ of $v_{n,o}$, and inductively keep v' good since only a finite number of values for $v'_{n,o}$ would cause v' to no longer be good.[13] □

For any $\epsilon > 0$ and $v \in V$, we can find $s \in S$ so that $||v - s|| < \epsilon$. By A.1, we get

$$|W_x(v, A) - W_x(s, A)| < \epsilon$$

and

$$|W_x(s, A) - W(v, x(s, A))| < \epsilon.$$

By the triangle inequality,

$$W_x(v, A) < W(v, x(s, A)) + 2\epsilon \leq \max_{r \in R} W(v, r) + 2\epsilon.$$

Since this is true for any $\epsilon > 0$,

$$W_x(v, A) \leq \max_{r \in R} W(v, r). \tag{11}$$

For any $r \in R$, let $s \in S$ be a value profile where $r = x(s, A)$. For any $v \in V$ we can construct v' as in A.2. Consider the profile v'' where

$$v''_{n,o} = \begin{cases} \max(s_{n,o}, v'_{n,o}) & x_n(s) = o \\ \min(s_{n,o}, v'_{n,o}) & x_n(s) \neq o \end{cases}$$

[13] Although the argument doesn't work if $V_{n,o}$ is a singleton, in this case we don't even need to change the value as $s_{n,o} = v_{n,o}$

Consider changing the value profile from s to v'' one value at a time. Since x is weakly monotone and each intermediate profile $v^* \in S$ by construction, the allocation along this path must always be equivalent to r. Therefore

$$W_x(v'', A) = W(v'', r).$$

By *A*.1, the welfare function is nondecreasing and 1-Lipschitz

$$W_x(v', A) \geq W_x(v'', A) - \sum_{x_n(s)=o} v''_{n,o} - v'_{n,o} = W(v', r).$$

Since we could construct v' for any $\epsilon > 0$ and the welfare function is continuous

$$W_x(v, A) \geq W(v, r).$$

Since this is true for all $r \in R$,

$$W_x(v, A) \geq \max_{r \in R} W(v, r). \tag{12}$$

Combining 11 and 12, we get

$$W_x(v, A) = \max_{r \in R} W(v, r),$$

so x is a quasi-optimizer.

Proof of Theorem 2.2

The certain investment instances are a subset of the investment instances. Thus, if x is a β-approximation for investment, then x is a β-approximation for certain investment.

We extend the relative utility notation u as follows:

$$u_\iota((v_\iota, c), v_{-\iota}, A, x) \equiv \max_{o \in O} \{v_\iota^o - \tau_\iota^o(v_{-\iota}, A, x)\} - c,$$

$$u_\iota(I, v_{-n}, A, x) \equiv \max_{(v_\iota, c) \in I} \max_{o \in O} u_\iota((v_\iota, c), v_{-\iota}, A, x).$$

For the other direction, first we construct a new set of investments I'. Let u^* be the expected relative utility of the set of investments I. Define

$$I' = \{(\nu_\iota, c + u^*) : (\nu_\iota, c) \in I\} \cup (\tau_\iota, 0).$$

Notice that we include $(\tau_\iota, 0)$ as an investment option to satisfy the requirement that one investment has 0 cost. By construction, for each $(\hat{\nu}_\iota, \hat{c}) \in \text{BR}(x, g, I, \nu_{-\iota}, \mathcal{A})$ we have $(\hat{\nu}_\iota, \hat{c} + u^*) \in \text{BR}(x, g, I', \nu_{-\iota}, \mathcal{A})$. For each $(\nu_\iota, c + u^*) \in I'$ we can define the cost function

$$\overline{c}(s) = u_\iota(v(s), \mathcal{A}(s), x) + (u^* - u_\iota((\nu_\iota, c), \nu_{-\iota}, \mathcal{A}, x))$$

that can be thought of as a cost in each state since $E(\overline{c}) = c + u^*$. Because $u_\iota((\nu_\iota, c), \nu_{-\iota}, \mathcal{A}, x) \leq u^*$, we have

$$u_\iota((\nu_\iota(s), \overline{c}(s)), \nu_{-\iota}(s), \mathcal{A}(s), x) \leq 0 \quad \forall s \in S \quad \forall (v_\iota, c + u^*) \in I'.$$

For each $(\hat{\nu}_\iota, \hat{c} + u^*) \in \text{BR}(x, g, I', \nu_{-\iota}, \mathcal{A})$[14] since $u_\iota((\hat{\nu}_\iota, \hat{c}), \nu_{-\iota}, \mathcal{A}, x) = u^*$, we have

$$u_\iota((\hat{\nu}_\iota(s), \overline{\hat{c}}(s)), \nu_{-\iota}(s), \mathcal{A}(s), x) = 0 \quad \forall s \in S.$$

Therefore $(\hat{\nu}_\iota(s), \overline{\hat{c}}(s))$ is a best response in each state.
Since x is a β-approximation for certain investment

$$\overline{W}_x(I', \mathcal{A}) \geq \beta E_g(\overline{W}^*(I'(s), \mathcal{A}(s))) \geq \beta \overline{W}^*(I', \mathcal{A}).$$

Therefore,

$$\overline{W}_x(I, \mathcal{A}) \geq \overline{W}_x(I', \mathcal{A}) + u^* \geq \beta(\overline{W}^*(I', \mathcal{A})) + u^* \geq \beta \overline{W}^*(I, \mathcal{A}) + (1 - \beta)u^* \geq \beta \overline{W}^*(I, \mathcal{A}),$$

since the first and third inequalities follow from the construction of I' and the final inequality holds because of the requirement that one investment has 0 cost, implying $u^* \geq 0$.

So x is a β-approximation for investment.

Proof of Theorem 2.3

To prove Theorem 2.3, we start with the following technical lemma.

Lemma A.3. *$W^*(v, A)$ is non-decreasing in v_n and 1-Lipschitz in v_n in the sup norm.*

Proof. $W^*(v, A)$ is non-decreasing by inspection. To establish it is 1-Lipschitz, take any two instances, (v_n, v_{-n}, A) and $(\tilde{v}_n, v_{-n}, A)$. Let a and $\tilde{a}$ be allocations that attain the maximum

[14] The following statement also holds for $(\tau_\iota, 0) \in \text{BR}(x, g, I', \nu_{-\iota}, \mathcal{A})$

at these two instances respectively. We have that

$$\begin{aligned}W^*(v_n, v_{-n}, A) &\geq v_n \cdot \tilde{a}_n + \sum_{m \neq n} v_m \cdot \tilde{a}_m \\ &\geq \tilde{v}_n \cdot \tilde{a}_n - \max_o \{|\tilde{v}_n^o - v_n|\} + \sum_{m \neq n} v_m \cdot \tilde{a}_m \\ &= W^*(\tilde{v}_n, v_{-n}, A) - \max_o \{|\tilde{v}_n^o - v_n|\}.\end{aligned}$$

By symmetry, it follows that the function is 1-Lipschitz. □

To show that β-pivotal is necessary, consider $I = \{(v_\iota, u_\iota(v_\iota, v_{-\iota})), (\tau_\iota(v_{-\iota}, A, x), 0)\}$. To ease notation, we define $t \equiv \tau_\iota(v_{-\iota}, A, x)$. We suppress the dependence of W_x, W^*, $\overline{W}_x$, $\overline{W}^*$, and u_ι on the arguments $v_{-\iota}, A$, since these arguments are held constant throughout the proof. Since the algorithm is a β-approximation for investment I and $(v_\iota, u_\iota(v_\iota, v_{-\iota}))$ is a best response

$$W_x(v_\iota) - u_\iota(v_\iota, x) \geq \beta W^*(t),$$

which is the β-pivotal condition.
To show β-pivotal is sufficient, let $(\hat{v}_\iota, \hat{c}_\iota) \in \text{BR}(x, I, v_{-\iota}, A)$ give the minimum total welfare so by definition

$$\overline{W}_x(I) = W_x(\hat{v}_\iota) - \hat{c}_\iota. \tag{13}$$

Since x is β-pivotal,

$$\begin{aligned}W_x(\hat{v}_\iota) &\geq \beta W^*(t) + u_\iota(\hat{v}_\iota, x) \\ &= \beta W^*(t) + u_\iota((\hat{v}_\iota, \hat{c}_\iota), x) + \hat{c}_\iota \\ &= \beta W^*(t) + u_\iota(I, x) + \hat{c}_\iota,\end{aligned}$$

where u_ι is defined as in the proof of Theorem 2.2. Rearranging we get

$$W_x(\hat{v}_\iota) - \hat{c}_\iota \geq \beta W^*(t) + u_\iota(I, x). \tag{14}$$

For any $(v'_\iota, c'_\iota) \in I$, since W^* is 1-Lipschitz and nondecreasing (by Lemma A.3),

$$W^*(t) + u_\iota(I, x) \geq W^*(t) + u_\iota((v'_\iota, c'_\iota), x) \geq W^*(v'_\iota) - c'_\iota.$$

This is true for all $(v'_\iota, c'_\iota) \in I$, so we get

$$W^*(t) + u_\iota(I, x) \geq \overline{W}^*(I). \tag{15}$$

Combining equations (13), (14), (15), and using the fact that $u_\iota(I,x) \geq 0$ (because one of the investments has 0 cost), we get the desired result

$$\overline{W}_x(I) = W_x(\hat{v}_\iota) - \hat{c}_\iota \geq \beta W^*(t) + u_\iota(I,x) \geq \beta \overline{W}^*(I).$$

Proof of Theorem 2.4

If x is β-pivotal, then by Theorem 2.3 x is a β-approximation for certain investment. Consider the investment technology

$$I \equiv \{(v_n, 0), (\tilde{v}_n, u_n(v,A,x) - u_n(\tilde{v}_n, v_{-n}, A, x))\}.$$

The cost is chosen so that $(\tilde{v}_n, u_n(v,A,x) - u_n(\tilde{v}_n, v_{-n}, A, x))$ is a best response. Since x is a β-approximation for certain investment,

$$W_x(\tilde{v}_n, v_{-n}, A, x) - u_n(v,A,x) + u_n(\tilde{v}_n, v_{-n}, A, x) \geq \beta W^*(v,A).$$

Subtracting $W_x(v,A)$ from both sides and using the definition of W_x

$$\begin{aligned} &u_n(\tilde{v}_n, v_{-n}, A, x) - u_n(v,A,x) + \sum_{m \in N} v_m \cdot [x_m(\tilde{v}_n, v_{-n}, A) - x_m(v,A)] \\ &\geq \beta W^*(v,A) - W_x(v,A). \end{aligned}$$

By Lemma 2.2,

$$\begin{aligned} \mathcal{E}_x(\tilde{v}_n, (v,A)) &= p_n(\tilde{v}_n, v_{-n}, A) - p_n(v,A) + \sum_{m \neq n} v_m \cdot [x_m(\tilde{v}_n, v_{-n}, A) - x_m(v,A)] \\ &\geq \beta W^*(v,A) - W_x(v,A). \end{aligned}$$

so x has β-bounded confirming externalities.

Assume x has β-bounded confirming externalities. For any v_n and $\epsilon \in (0,1]$, v_n confirms $(v_n^\epsilon, v_{-n}, A)$ for algorithm x where

$$v_n^\epsilon = \epsilon v_n + (1-\epsilon)\tau_n(v_{-n}, A, x).$$

Since x has β-bounded confirming externalities,

$$p_n(v, A) - p_n(v_n^\epsilon, v_{-n}, A) + \sum_{m \neq n} v_m \cdot [x_m(v, A) - x_m(v_n^\epsilon, v_{-n}, A)]$$
$$= \mathcal{E}_x(v_n, (v_n^\epsilon, v_{-n}, A)) \geq \beta W^*(v_n^\epsilon, v_{-n}, A) - W_x(v_n^\epsilon, v_{-n}, A).$$

By Lemma 2.2,

$$W_x(v, A) - W_x(v_n^\epsilon, v_{-n}, A) - u_n(v, A, x) + u_n(v_n^\epsilon, v_{-n}, A, x) \geq \beta W^*(v_n^\epsilon, v_{-n}, A) - W_x(v_n^\epsilon, v_{-n}, A).$$

Canceling $W_x(v_n^\epsilon, v_{-n}, A)$ on both sides and taking the limit as $\epsilon \to 0$, we get

$$W_x(v, A) - u_n(v, A, x) \geq \beta W^*(\tau_n(v_{-n}, A, x), v_{-n}, A).$$

Therefore x is β-pivotal.

Proof of Proposition 2.5

Suppose that y and X satisfy the assumptions of Proposition 2.5. We want to prove that for any (v, A) and $\tilde{v}_n$,

$$0 \leq [\tilde{v}_n - v_n] \cdot [y_n(\tilde{v}_n, v_{-n}, A) - y_n(v, A)]. \tag{16}$$

To ease notation we hold A fixed and suppress the dependence of functions on A. If $y_n(\tilde{v}_n, v_{-n}) = y_n(v)$ then (16) follows immediately. Suppose $y_n(\tilde{v}_n, v_{-n}) \neq y_n(v)$. If the change from v_n to $\tilde{v}_n$ confirms $y_n(\tilde{v}_n, v_{-n})$ then (16) follows immediately. Suppose it does not confirm $y_n(\tilde{v}_n, v_{-n})$. Then by $y_n(\tilde{v}_n, v_{-n}) \neq y_n(v)$ and $|O| = 2$, the change from v_n to $\tilde{v}_n$ confirms $y_n(v)$, and the change from $\tilde{v}_n$ to v_n confirms $y_n(\tilde{v}_n, v_{-n})$.

Let us pick $x, \tilde{x} \in X$ such that $x(v) = y(v)$ and $\tilde{x}(v) = y(\tilde{v}_n, v_{-n})$. We have

$$\begin{aligned} v_n \cdot \tilde{x}_n(v) + \sum_{m \neq n} v_m \cdot \tilde{x}(v) &\leq v_n \cdot x_n(v) + \sum_{m \neq n} v_m \cdot x_m(v) \\ &\leq v_n \cdot x_n(\tilde{v}_n, v_{-n}) + \sum_{m \neq n} v_m \cdot x_m(\tilde{v}_n, v_{-n}), \end{aligned} \tag{17}$$

where the first inequality is by construction and the second inequality is by x XCONE and weakly monotone and Proposition 2.2. A symmetric argument yields

$$\begin{aligned}\tilde{v}_n \cdot x_n(\tilde{v}_n, v_{-n}) + \sum_{m\neq n} v_m \cdot x_m(\tilde{v}_n, v_{-n}) &\leq \tilde{v}_n \cdot \tilde{x}_n(\tilde{v}_n, v_{-n}) + \sum_{m\neq n} v_m \cdot \tilde{x}_m(\tilde{v}_n, v_{-n}) \\ &\leq \tilde{v}_n \cdot \tilde{x}_n(v) + \sum_{m\neq n} v_m \cdot \tilde{x}_m(v). \end{aligned} \tag{18}$$

Adding inequalities (17) and (18) and canceling terms yields

$$0 \leq [\tilde{v}_n - v_n] \cdot [\tilde{x}_n(v) - x_n(\tilde{v}_n, v_{-n})]. \tag{19}$$

Since the change from v_n to $\tilde{v}_n$ confirms $y_n(v) = x_n(v)$, and x_n is weakly monotone, we have

$$0 = [\tilde{v}_n - v_n] \cdot [x_n(\tilde{v}_n, v_{-n}) - x_n(v)]. \tag{20}$$

Similarly, since the change from $\tilde{v}_n$ to v_n confirms $y_n(\tilde{v}_n, v_{-n}) = \tilde{x}_n(\tilde{v}_n, v_{-n})$, and $\tilde{x}_n$ is weakly monotone, we have

$$0 = [\tilde{v}_n - v_n] \cdot [\tilde{x}_n(\tilde{v}_n, v_{-n}) - \tilde{x}_n(v)]. \tag{21}$$

Adding (19), (20), and (21) yields

$$0 \leq [\tilde{v}_n - v_n] \cdot [\tilde{x}_n(\tilde{v}_n, v_{-n}) - x_n(v)] = [\tilde{v}_n - v_n] \cdot [y_n(\tilde{v}_n, v_{-n}) - y_n(v)],$$

as desired.

B Steiner tree

The Steiner tree problem is a classic problem with applications in package delivery and optimal routing. The input to the problem is a connected, undirected graph $G = (V, E)$, where each edge has a weight, and a set $V^* \subseteq V$ of nodes selected as *terminals*. The goal is to find a weight-minimizing connected subgraph of G which contains all the terminals. It is well-known that this problem is NP-Complete (Karp, 1972).

The Steiner tree problem has a classic 2-approximation that builds on the minimum spanning tree (MST) of the graph G (Vazirani, 2013, pp. 27–28). The MST-based 2-approximation algorithm for the Steiner tree problem works in three steps:

Algorithm 3 (MST-Steiner).
1. Construct a weighted graph G' from the original graph G in the following way: The set of nodes of G' are the terminal nodes of G. For any two nodes t_1 and t_2 in G', let the weight of the edge between them be equal to the

total weight of the shortest path between the two in G.

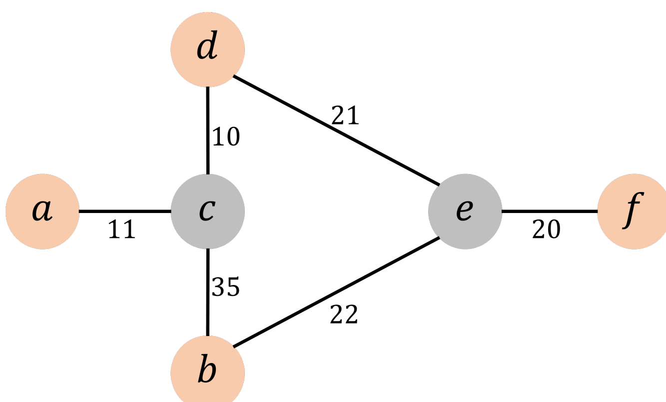
2. Find a minimum spanning tree in G'.

3. Recover the shortest paths in the original graph G that represent the edges in the minimum spanning tree just constructed; then remove edges if necessary to ensure that the output is a tree.

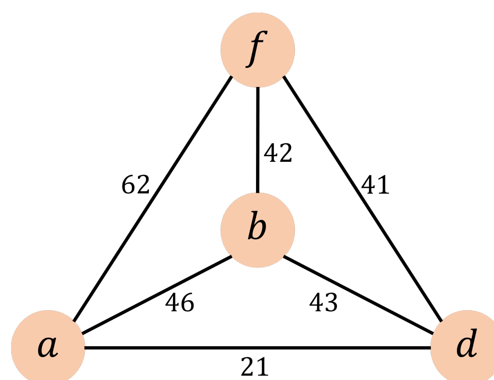
We have stated XCONE for value-maximization problems, but it naturally generalizes to cost-minimization problems such as this one, which can be interpreted as procurement auctions. In particular, each edge in G is a bidder, each bidder's value is the negative of the weight, and the packed bidders are the edges that are selected by the algorithm.

Proposition B.1. MST-Steiner *is not XCONE.*

Proof. We prove Proposition B.1 by example. The graph in Figure 1a represents the initial graph G. Nodes $\{a, b, d, f\}$ are the terminal nodes. The graph in Figure 1b is G', which we have constructed from G. The MST of G' includes edges ad, df, and bf. These correspond to ac, cd, de, ef, and be, which together comprise a Steiner tree in the original graph G. The total weight of this Steiner tree is 76.

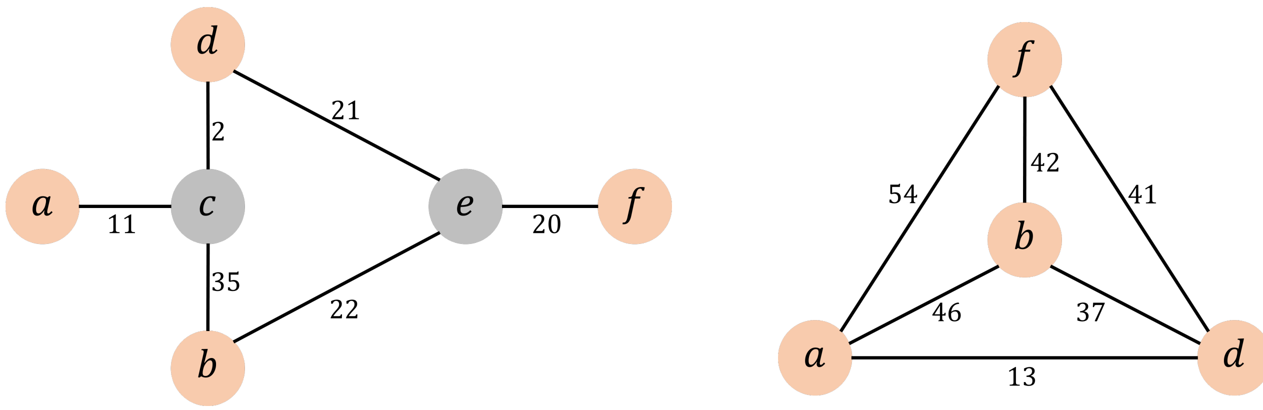


(a) Graph G



(b) Graph G' constructed from G

Figure 1: Steiner tree instance before investment



(a) Post-investment graph G

(b) Graph G' constructed from post-investment G

Figure 2: Steiner tree instance after investment

Now suppose we introduce an investment that reduces the weight of cd from 10 to 2 as pictured in Figure 2a (this is equivalent to *increasing* the value of that edge in the corresponding maximization problem). Applying the same algorithm (as illustrated in Figure 2b) leads to choosing ac, cd, bc, de, and ef, with a total weight of 91. In particular, we reduced the weight of a packed edge cd, and the total weight incurred from the edges other than cd has risen by 13, which is a bossy negative externality. Thus, MST-Steiner is not XCONE. □