SUFFICIENCY OF WEAK MONOTONICITY FOR IMPLEMENTABILITY IN MAX DOMAINS*

Satish Agarwal[1] and Souvik Roy[2]

[1]Reservoir Engineer in Total E&P, Myanmar

[2]Corresponding Author, Economic Research Unit, Indian Statistical Institute, Kolkata

Abstract

We consider non-convex domains where the valuation of a bundle of objects is given by the maximum valuation of the objects in that bundle. We show that an allocation rule is implementable if and only if it satisfies a familiar and simple condition called weak monotonicity or 2-cycle monotonicity.

Keywords: weak monotonicity; implementability; max domains;

JEL Classification: D44, D47, D71, D82, D86.

1 Introduction

An important research area in mechanism design is to provide simple sufficient conditions for implementability. An allocation rule is implementable if there exists a payment rule such that truth-telling is a dominant strategy for the agents in the resulting mechanism. Rochet (1987) shows that an allocation rule is implementable on any domain (of types) if and only if it satisfies k-cycle monotonicity for every integer $k \geq 2$ (see also Rockafellar (1970)). k-cycle monotonicity requires that for every sequence of types $t_1, \ldots, t_k$, if the outcome of an allocation rule is a_l when an agent i reports his/her type as t_l for all $l = 1, \ldots, k$ (where the types of all other agents are fixed), then $t_1(a_1) + t_2(a_2) + \cdots + t_{k-1}(a_{k-1}) + t_k(a_k) \geq t_1(a_2) + t_2(a_3) + \cdots + t_{k-1}(a_{k-1}) + t_k(a_1)$.

Clearly, checking whether an allocation rule satisfies k-cycle monotonicity for all integers $k \geq 2$ is a tedious job. This raises the question whether this condition can be further simplified for specific domains.

*The authors would like to gratefully acknowledge Debasis Mishra and Rudolf Muller for their insightful comments. This article is present on a website and can be accessed on http://www.isid.ac.in/~epu/acegd2015/papers/SouvikRoy.pdf. This article is not published nor is under publication elsewhere.

Myerson (1981) shows that in the single object auction set-up 2-cycle monotonicity is necessary and sufficient for implementation (see also Spence (1974), Mirrlees (1976)). Later, Bikhchandani et al. (2006), Saks and Yu (2005), Ashlagi et al. (2010) establish the same result for finite dimensional convex domains.

One important objective of mechanism design is to characterize mechanisms that maximize the expected revenue. Myerson (1981) solves this problem for the sale of a single object, however the same is not solved for the sale of multiple objects (see a recent take on this topic in Manelli and Vincent (2007), Hart and Nisan (2012), Hart and Reny (2012), and Daskalakis et al. (2017)). As mentioned by Myerson, it is very important to have a characterization of implementable allocation rules in order to find the revenue maximizing mechanism. Therefore, it is worth investigating whether 2-cycle monotonicity characterizes the implementable rules in a domain or not.

In this paper, we consider a particular type of domains called max domains. A max domain is one where an agent values a bundle of objects by the object in the bundle that is most valuable for him/her. In other words, objects are perfectly substitutable in this setting. Max domains can be seen as a special case of a general class of domains called gross substitute domains (Gul and Stacchetti (1999), Danilov et al. (2003)). These domains occur in both public and private good allocation problems.

Public good allocation problems: Consider the problem of allocating some identical public goods. For instance, suppose that the social planner has to choose some locations in a city to build up some schools (or some hospitals or bus-stops, etc). Since an agent will eventually use his/her most convenient location for the public good, this leads to a max domain.

Private good allocation problems: Consider an auction where 'similar' objects, say computers, laptops, tabs, etc., are to be sold. Suppose that the designer wants to get rid of the objects due to a shortage of storage space.[1] Note that these objects are substitutes, and hence an agent will practically use exactly one of these (even if he receives more than one). In other words, agents have unit demand, which leads to a max domain.

The main contribution of this paper is the result that implementability is equivalent to 2-cycle monotonicity in max domains. Max domains are not new to the literature. Vohra (2011) discusses these domains and shows that 2-cycle monotonicity implies 3-cycle monotonicity when there are exactly two objects. However, to the best of our knowledge, the question whether 2-cycle monotonicity implies cycle monotonicity for arbitrary number of objects remained open.

Since max domains are connected, it follows from Chung and Olszewski (2007) that revenue equivalence holds on these domains. By means of this fact, one can characterize the set of payments that implement a 2-cycle monotone allocation rule.

[1]This is a common scenario in mechanism design (particularly in case of division problems) where free disposal is not assumed.

The closest papers to ours are Mishra and Roy (2013) and Mishra et al. (2014). Mishra and Roy (2013) consider a non-convex domain called rich dichotomous domain and show that 3-cycle monotonicity is sufficient for implementability on that domain, but 2-cycle monotonicity is not sufficient. Mishra et al. (2014) show that 2-cycle monotonicity is sufficient for implementability on multidimensional single-peaked domains.

The proof technique of this paper is totally new and flexible. It uses the basic structure of the domains–not any advanced mathematical tool. We find the proof of the same result for convex domains in Bikhchandani et al. (2006) quite technical. The proof technique in Mishra et al. (2014) for single-peaked domains relies on the fact that valuations of two particular objects can be changed adequately keeping those of the other objects intact. However, such flexibility is not there in case of max domains. Recently, Kushnir and Lokutsievskiy (2019) show that 2-cycle monotonicity implies cycle monotonicity on gross-substitute (and some other) domains. They use Nerve theorem from algebraic topology to show this. Even though a max domain is a strict subset of a gross-substitute domain, their result or the proof technique (in particular, the Nerve theorem) does not apply to max domains. In other words, as in the case of Mishra et al. (2014), their proof technique too uses the full freedom of gross substitute domains which limits its applicability.

1.1 Related literature

In the context of Bayesian implementation with randomization, Jehiel et al. (1999) provide an integral condition that is required together with 2-cycle monotonicity in order to ensure implementability. Some other papers which provide technical conditions for implementability in different environments are Jehiel et al. (1999), Muller et al. (2007), Archer and Kleinberg (2008), Berger et al. (2010), and Carbajal and Ely (2013), Rahman (2011). The following papers provide spaces where revenue equivalence result in Myerson (1981) holds: Krishna and Maenner (2001), Milgrom and Segal (2002), Chung and Olszewski (2007), Heydenreich et al. (2009), Carbajal (2010), Kos and Messner (2013).

2 Model

We consider a set $N = \{1, \dots, n\}$ of n agents and a set $S = \{a_1, \dots, a_m\}$ of m objects. The set of alternatives $\mathcal{A}$ is defined as the set of all subsets of S. Thus, there are 2^m alternatives. A type is a vector in $\mathbb{R}^{2^m}$ that specifies the utilities of an agent for all alternatives. A domain is a collection of admissible types. An allocation rule f is a function from $\mathcal{D}^n$ to $\mathcal{A}$.

Definition 2.1 (Max Domain)**.** A domain $\mathcal{D}$ is called a max domain if $\mathcal{D} = \{t \in \mathbb{R}^{2^m} : t(X) = \max_{a \in X} t(\{a\}) \;\; \forall X \in \mathcal{A}\}$.

Thus, in a max domain, the utility of an agent from a bundle of objects is given by the maximum utility of that agent over the objects in that bundle.

Notation. For ease of presentation, we introduce the following notation: $\tau(X, t_i) = \{x \in X : t_i(x) \geq t_i(y) \text{ for all } y \in X\}$ where $X \subseteq S$ and $t_i \in \mathbb{R}^{2^m}$.

Definition 2.2. An allocation rule $f : \mathcal{D}^n \to \mathcal{A}$ satisfies k-cycle monotonicity for some integer $k \geq 2$ if for all $i \in N$, all $t_1, t_2, \ldots, t_k \in \mathcal{D}$, and all $t_{-i} \in \mathcal{D}^{n-1}$, we have

$$\sum_{j=1}^{k} t_j(f(t_j, t_{-i})) - t_j(f(t_{j+1}, t_{-i})) \geq 0,$$

where $t_{k+1} \equiv t_1$.

An allocation rule is said to satisfy weak monotonicity if it satisfies 2-cycle monotonicity.

Definition 2.3. An allocation rule $f : \mathcal{D}^n \to \mathcal{A}$ is implementable (quasi-linear utility environment) if there exists a payment function $p : \mathcal{D}^n \to \mathbb{R}$ such that for all $i \in N$, all $t_i, t_i' \in \mathcal{D}$, and all $t_{-i} \in \mathcal{D}^{n-1}$

$$t_i\left(f(t_i, t_{-i})\right) - p(t_i, t_{-i}) \geq t_i\left(f(t_i', t_{-i})\right) - p(t_i', t_{-i}).$$

3 Results

In this section we present the main result of this paper. First, we present a theorem due to Rockafellar (1970) that states that k-cycle monotonicity for all integers $k \geq 2$ is necessary and sufficient for implementability.

Theorem 3.1. *[Rockafellar (1970)] An allocation rule f is implementable if and only if it is k-cycle monotone for all integers $k \geq 2$.*

The main result in this paper says that if an allocation function on a max domain satisfies 2-cycle monotonicity, then it will satisfy k-cycle monotonicity for all integers $k \geq 2$. In view of Theorem 3.1, this implies that 2-cycle monotonicity is necessary and sufficient for implementability in max domains.

Theorem 3.2. *Weak monotonicity is sufficient for implementability in max domain.*

The proof of the theorem follows from the following lemma.

Lemma 3.1. *Let $k \geq 3$ be an integer and let $\mathcal{D}$ be a max domain. If an allocation rule $f : \mathcal{D}^n \to \mathcal{A}$ satisfies $(k-1)$-cycle monotonicity, then it satisfies k-cycle monotonicity.*

Proof. Suppose not. Then there exists an allocation rule $f : \mathcal{D}^n \to \mathcal{A}$, $k \geq 3$ types $t_1, \ldots, t_k \in \mathcal{D}$ of an agent $i \in N$, and a type profile $t_{-i} \in \mathcal{D}^{n-1}$ of all agents except i such that f violates k-cycle monotonicity over the type profiles $(t_1, t_{-i}), \ldots, (t_k, t_{-i})$. Let $f(t_j, t_{-i}) = A_j$ for all $j = 1, \ldots, k$. To simplify the presentation of the proof, we assume that $t_l(A_1) = \alpha$ for all $l = 1, \ldots, k$. Note that this does not affect the logic of our proof since constant terms can cancel from both sides of the (in)equations that we consider here.

Applying $(k-1)$-cycle monotonicity to the types $t_1, \ldots, t_{k-1}$ and using the assumption that $t_1(A_1) = t_{k-1}(A_1)$, we have

$$t_2(A_2) + t_3(A_3) + \cdots + t_{k-1}(A_{k-1}) \geq t_1(A_2) + t_2(A_3) + \cdots + t_{k-2}(A_{k-1}). \tag{1}$$

Again, applying $(k-1)$-cycle monotonicity to the types $t_2, \ldots, t_k$, we have

$$t_2(A_2) + t_3(A_3) + \cdots + t_k(A_k) \geq t_2(A_3) + \cdots + t_{k-1}(A_k) + t_k(A_2). \tag{2}$$

Since f violates k-cycle monotonicity over the types $t_1, \ldots, t_k$ and $t_1(A_1) = t_k(A_1)$ by our assumption, we have

$$t_2(A_2) + t_3(A_3) + \cdots + t_k(A_k) < t_1(A_2) + t_2(A_3) + \cdots + t_{k-1}(A_k). \tag{3}$$

Note that if $t_k(A_k) \geq t_{k-1}(A_k)$, then (1) contradicts (3). Moreover, if $t_k(A_2) \geq t_1(A_2)$, then (2) contradicts (3). Suppose $\tau(A_2, t_1) = \tau(A_k, t_{k-1})$. This means $\tau(A_2, t_1) \in A_k$, and hence we have $t_1(A_k) \geq t_1(A_2)$. Similarly, we have $t_{k-1}(A_2) \geq t_{k-1}(A_k)$. Now, applying 2-cycle monotonicity to the types t_1 and t_k, we obtain $t_k(A_k) \geq t_1(A_k)$. This, along with the fact that $t_1(A_k) \geq t_1(A_2)$, implies $t_k(A_k) \geq t_1(A_2)$. Applying $(k-2)$-cycle monotonicity to the types $t_2, \ldots, t_{k-1}$, we have

$$t_2(A_2) + t_3(A_3) + \ldots + t_{k-1}(A_{k-1}) \geq t_2(A_3) + \ldots + t_{k-2}(A_{k-1}) + t_{k-1}(A_2). \tag{4}$$

Using the facts that $t_{k-1}(A_2) \geq t_{k-1}(A_k)$ and $t_k(A_k) \geq t_1(A_2)$ in (4), we have

$$t_2(A_2) + t_3(A_3) + \cdots + t_{k-1}(A_{k-1}) + t_k(A_k) \geq t_1(A_2) + t_2(A_3) + \cdots + t_{k-1}(A_k),$$

which violates (3). Note that if $k = 3$, then we do not need (4) to arrive at this contradiction.

Now, we consider the case where the following conditions are satisfied. Note that this is the only remaining case.

$$t_{k-1}(A_k) > t_k(A_k), \tag{5}$$

$$t_1(A_2) > t_k(A_2), \text{ and} \tag{6}$$

$$\tau(A_2, t_1) \neq \tau(A_k, t_{k-1}). \tag{7}$$

Take $b, c \in S$ such that $b \neq c$, $b \in \tau(A_2, t_1)$ and $c \in \tau(A_k, t_{k-1})$. Note that such a choice is possible by assumption (7). Consider the type t_{k+1} such that

$$\begin{aligned} t_{k+1}(\{b\}) &= t_1(\{b\}) - \epsilon, \\ t_{k+1}(\{c\}) &= t_k(A_k) + \epsilon, \text{ and} \\ t_{k+1}(\{x\}) &= t_k(\{x\}) \text{ for all } x \in S \setminus \{b, c\} \end{aligned}$$

where $\epsilon > 0$ is arbitrarily small.

To help the reader, we illustrate the construction of t_{k+1} for $k = 3$ by means of the following example. The circled numbers in Table 3 indicate the fact that when agent i has the type given in the corresponding row (where other agents have some fixed types), the outcome is the alternative at the corresponding column. Here, ϵ is chosen as 0.3 for t_4.

	a	b	c	$max(ab)$	$max(bc)$	$max(ca)$	$max(abc)$
t_1	1	3	②	3	3	1	3
t_2	5	3	2	5	③	5	5
t_3	4	1	2	4	2	④	4
t_4	4.3	2.7	2	4.3	2.7	4.3	4.3

Table 1: Construction of t_{k+1} for $k = 3$

Claim 3.1. *For the type t_{k+1}*

$$t_{k+1}(X) \geq t_k(X) \textit{ for all } X \in \mathcal{A}. \tag{8}$$

Proof. By the construction of t_{k+1}, it is enough to show that $t_{k+1}(\{c\}) > t_k(\{c\})$ and $t_{k+1}(\{b\}) > t_k(\{b\})$. Because $c \in A_k$, we have $t_k(A_k) \geq t_k(\{c\})$, and hence $t_{k+1}(\{c\}) = t_k(A_k) + \epsilon > t_k(\{c\})$. Since $b \in \tau(A_2, t_1)$, we have $t_1(\{b\}) = t_1(A_2)$. Hence, by (6), we have $t_1(\{b\}) > t_k(A_2) \geq t_k(\{b\})$. Here, the last inequality follows from the fact that $b \in A_2$. Since ϵ is arbitrarily small, we obtain $t_{k+1}(\{b\}) = t_1(\{b\}) - \epsilon > t_k(\{b\})$, which completes the proof of Claim 3.1.

Claim 3.2. *It must be that $t_{k+1}(A_1) = \alpha$.*

Proof. Note that by the construction of t_{k+1}, $t_{k+1}(A_1) = t_k(A_1) = \alpha$ if $b, c \notin A_1$. We first show that $b \notin \tau(A_1, t_{k+1})$. If $b \notin A_1$, then there is nothing to prove. Suppose $b \in A_1$. This means either $t_1(A_1) \geq t_1(\{b\})$ or $t_1(A_1) > t_1(\{b\}) - \epsilon = t_{k+1}(\{b\})$. Since $t_1(A_1) = t_k(A_1) = \alpha$, Claim 3.1 implies $t_{k+1}(A_1) \geq t_k(A_1) > t_{k+1}(\{b\})$. This proves that $b \notin \tau(A_1, t_{k+1})$.

Now, we show $c \notin \tau(A_1, t_{k+1})$. As before, if $c \notin A_1$, then there is nothing to prove. Suppose $c \in A_1$. This means $t_{k-1}(A_1) \geq t_{k-1}(\{c\})$. Since $c \in \tau(A_k, t_{k-1})$, we have $t_{k-1}(\{c\}) = t_{k-1}(A_k)$. This implies $t_{k-1}(A_1) \geq t_{k-1}(A_k)$. Using (5), this yields $t_{k-1}(A_1) > t_k(A_k)$. Since ϵ is arbitrarily small, we have $t_{k-1}(A_1) > t_k(A_k) + \epsilon = t_{k+1}(c)$. Moreover, as $t_{k-1}(A_1) = t_k(A_1) = \alpha$, we have $t_k(A_1) > t_{k+1}(c)$. Now, using Claim 3.1, we get $t_{k+1}(A_1) \geq t_k(A_1) > t_{k+1}(\{c\})$. This proves $c \notin \tau(A_1, t_{k+1})$, completing the proof of Claim 3.2.

We now complete the proof of Lemma 3.1 by showing that f violates $(k-1)$ or lower cycle monotonicity for every possible outcome at t_{k+1}. Note that for any alternative $X \in \mathcal{A}$,

$$\begin{aligned} t_{k+1}(X) &= t_k(X) \text{ if } b, c \notin \tau(X, t_{k+1}), \text{ and} \\ t_{k+1}(X) &\in \{t_1(A_2) - \epsilon, t_k(A_k) + \epsilon\} \text{ otherwise.} \end{aligned}$$

In what follows, we consider all the above possibilities case by case.

Case 1. Consider an alternative X such that $t_{k+1}(X) = t_k(X)$. Suppose $f(t_{k+1}, t_{-i}) = X$. Note that since $c \in A_k$, we have $t_{k+1}(A_k) \geq t_{k+1}(\{c\}) = t_k(A_k) + \epsilon > t_k(A_k)$, and hence

$$t_{k+1}(X) + t_k(A_k) < t_k(X) + t_{k+1}(A_k).$$

However, this implies that f violates 2-cycle monotonicity over the types t_k and t_{k+1}, a contradiction.

Case 2. Consider an alternative X such that $t_{k+1}(X) \neq t_k(X)$. Note that then $t_{k+1}(X)$ is either $t_1(A_2) - \epsilon$ or $t_k(A_k) + \epsilon$. We distinguish the two cases based on whether $t_1(A_2) - \epsilon \neq t_k(A_k) + \epsilon$ or $t_1(A_2) - \epsilon = t_k(A_k) + \epsilon$.

Case 2.1. Suppose $t_1(A_2) - \epsilon \neq t_k(A_k) + \epsilon$.

Suppose further that $t_{k+1}(X) = t_1(A_2) - \epsilon$. This implies $b \in X$. As $b \in \tau(A_2, t_1)$, we have $t_1(X) \geq t_1(A_2)$. Suppose $f(t_{k+1}, t_{-i}) = X$. Then,

$$t_{k+1}(X) + t_1(A_1) < t_1(A_2) + t_{k+1}(A_1) \leq t_1(X) + t_{k+1}(A_1),$$

which implies that f violates 2-cycle monotonicity over t_1 and t_{k+1}.

Now, suppose $t_{k+1}(X) = t_k(A_k) + \epsilon$. This means $c \in X$. Applying $(k-1)$-cycle monotonicity to the types $t_2, \ldots, t_{k-1}, t_{k+1}$, we have

$$t_2(A_2) + t_3(A_3) + \ldots + t_{k-1}(A_{k-1}) + t_{k+1}(X) \geq t_{k+1}(A_2) + t_2(A_3) + \ldots + t_{k-1}(X). \tag{9}$$

As $b \in \tau(A_2, t_1)$, we have $t_{k+1}(A_2) \geq t_{k+1}(\{b\}) = t_1(\{b\}) - \epsilon = t_1(A_2) - \epsilon$. Moreover, as $c \in X$ and $c \in \tau(A_k, t_{k-1})$, we have $t_{k-1}(X) \geq t_{k-1}(c) = t_{k-1}(A_k)$. Using all these observations in (9), we have

$$t_2(A_2) + t_3(A_3) + \ldots + t_{k-1}(A_{k-1}) + t_k(A_k) + \epsilon \geq t_1(A_2) - \epsilon + t_2(A_3) + \ldots + t_{k-1}(A_k).$$

Since ϵ is arbitrarily small, this violates (3), a contradiction.

Case 2.2. Suppose that $t_{k+1}(X) = t_1(A_2) - \epsilon = t_k(A_k) + \epsilon$. This implies either b or c is in X. Hence, by Case 2.1, we have $f(t_{k+1}, t_{-i}) \neq X$.

Thus, it follows that f violates $(k-1)$ or lower cycle monotonicity for every possible outcome at t_{k+1}, which violates the assumption of Lemma 3.1. This completes the proof of Lemma 3.1. ■

Remark 1. (*Revenue equivalence*). An allocation rule $f : \mathcal{D}^n \to \mathcal{A}$ satisfies revenue equivalence if for all payment rules p, q that implement f, there exists a constant $\alpha \in \mathbb{R}$ such that for all $t \in \mathcal{D}^n$, $p(t) = q(t) + \alpha$. Since max domain is a connected subset of $\mathbb{R}^{2^m}$, it follows from Chung and Olszewski (2007) that the revenue equivalence property holds in these domains. There are standard methods in the literature to construct a payment function p for an allocation function f so that p implements f.

4 Conclusion

We have considered a particular type of non-convex domains which we call max domains. These domains occur in both public and private good settings. We have shown that 2-cycle monotonicity is necessary and sufficient for implementability in these domains. An important open problem in this area is to explore the cycle-monotonicity property of domains that are not convex but union of several convex domains.

References

[1] A. Archer and R. Kleinberg. Truthful germs are contagious: A local to global characterization of truthfulness. In *In Proceedings of the 9th ACM conference on Electronic commerce (EC-08)*. Springer (Lecture Notes in Computer Science), 2008.

[2] Itai Ashlagi, Mark Braverman, Avinatan Hassidim, and Dov Monderer. Monotonicity and Implementability. *Econometrica*, 78:1749–1772, 2010.

[3] André Berger, Rudolf Müller, and Seyed Hossein Naeemi. Path-montonicity and incentive compatibility. Working Paper, Maastricht University, 2010.

[4] S. Bikhchandani, S. Chatterji, R. Lavi, A. Mualem, N. Nisan, and A. Sen. Weak monotonicity characterizes deterministic dominant strategy implementation. *Econometrica*, 74:1109–1132, 2006.

[5] Juan Carlos Carbajal. On the uniqueness of groves mechanisms and the payoff equivalence principle. *Games and Economic Behavior*, 68(2):763–772, 2010.

[6] Juan Carlos Carbajal and Jeffrey C Ely. Mechanism design without revenue equivalence. *Journal of Economic Theory*, 148(1):104–133, 2013.

[7] Kim-Sau Chung and Wojciech Olszewski. A non-differentiable approach to revenue equivalence. *Theoretical Economics*, 2:1–19, 2007.

[8] Vladimir I Danilov, Gleb A Koshevoy, and Christine Lang. Gross substitution, discrete convexity, and submodularity. *Discrete Applied Mathematics*, 131(2):283–298, 2003.

[9] Constantinos Daskalakis, Alan Deckelbaum, and Christos Tzamos. Strong duality for a multiple-good monopolist. *Econometrica*, 85(3):735–767, 2017.

[10] Faruk Gul and Ennio Stacchetti. Walrasian equilibrium with gross substitutes. *Journal of Economic theory*, 87(1):95–124, 1999.

[11] S Hart and N Nisan. Approximate revenue maximization with multiple items. the hebrew university of jerusalem. *Center for Rationality DP-606*, 2012.

[12] Sergiu Hart and Philip J Reny. Maximizing revenue with multiple goods: Nonmonotonicity and other observations. hebrew university of jerusalem. *Center for Rationality DP-630*, 2012.

[13] B. Heydenreich, Rudolf Muller, M. Uetz, and R. V. Vohra. Characterization of revenue equivalence. *Econometrica*, 77:307–316, 2009.

[14] Philippe Jehiel, Benny Moldovanu, and Ennio Stacchetti. Multidimensional mechanism design for auctions with externalities. *Journal Economic Theory*, 85:258–293, 1999.

[15] Nenad Kos and Matthias Messner. Extremal incentive compatible transfers. *Journal of Economic Theory*, 148(1):134–164, 2013.

[16] Vijay Krishna and Eliot Maenner. Convex potentials with an application to mechanism design. *Econometrica*, 69:1113–1119, 2001.

[17] Alexey I Kushnir and Lev Lokutsievskiy. On the equivalence of weak-and cyclic-monotonicity. *Available at SSRN 3422846*, 2019.

[18] A. M. Manelli and D. Vincent. Multidimensional mechanism design: Revenue maximization and the multiple good monopoly. *Journal of Economic Theory*, 137:153–185, 2007.

[19] P. Milgrom and I. Segal. Envelope theorems for arbitrary choice sets. *Econometrica*, 70:583–601, 2002.

[20] James A Mirrlees. Optimal tax theory: A synthesis. *Journal of public Economics*, 6(4):327–358, 1976.

[21] Debasis Mishra and Souvik Roy. Implementation in multidimensional dichotomous domains. *Theoretical Economics*, 8:431–466, 2013.

[22] Debasis Mishra, Anup Pramanik, and Souvik Roy. Multidimensional mechanism design in single peaked type spaces. *Journal of Economic Theory*, 153:103–116, 2014.

[23] Rudolf Muller, Andres Perea, and Sascha Wolf. Weak monotonicity and bayes-nash incentive compatibility. *Games and Economics Behavior*, 61:344–358, 2007.

[24] Roger B. Myerson. Optimal auction design. *Mathematics of Operations Research*, 6:58–73, 1981.

[25] David Rahman. Detecting profitable deviations. Working Paper, University of Minnesota, 2011.

[26] J. C. Rochet. A necessary and sufficient condition for rationalizability in a quasi-linear context. *Journal of Mathematical Economics*, 16:191–200, 1987.

[27] R. T. Rockafellar. *Convex Analysis*. Princeton University Press, 1970.

[28] M. E. Saks and L. Yu. Weak Monotonicity Suffices for Truthfulness on Convex Domains. In *Proceedings of* 7^{th} *ACM Conference on Electronic Commerce*, pages 286–293. ACM Press, 2005.

[29] Michael Spence. Competitive and optimal responses to signals: An analysis of efficiency and distribution. *Journal of Economic theory*, 7(3):296–332, 1974.

[30] Rakesh V Vohra. *Mechanism design: a linear programming approach*, volume 47. Cambridge University Press, 2011.