Theoretical Economics 8 (2013), 431–466 1555-7561/20130431

Implementation in multidimensional dichotomous domains

Debasis Mishra
Economics and Planning Unit, Indian Statistical Institute

Souvik Roy
Economics and Planning Unit, Indian Statistical Institute

We consider deterministic dominant strategy implementation in multidimensional *dichotomous* domains in a private values and quasilinear utility setting. In such multidimensional domains, an agent's type is characterized by a single number, the value of the agent, and a nonempty set of acceptable alternatives. Each acceptable alternative gives the agent utility equal to his value and other alternatives give him zero utility. We identify a new condition, which we call generation monotonicity, that is necessary and sufficient for implementability in any dichotomous domain. If such a domain satisfies a richness condition, then a weaker version of generation monotonicity, which we call 2-generation monotonicity (equivalent to 3-cycle monotonicity), is necessary and sufficient for implementation. We use this result to derive the optimal mechanism in a one-sided matching problem with agents who have dichotomous types.

Keywords. Dominant strategy implementation, cycle monotonicity, dichotomous preferences, generation monotonicity.

JEL classification. C78, C79, D02, D44.

1. Introduction

We study multidimensional mechanism design in private value and quasilinear environments, e.g., auction domains, matching problems with transfers, and choosing a public good among multiple public goods with transfers. We restrict attention to deterministic implementation in dominant strategies. Our focus is on domains where agents have *dichotomous* preferences over alternatives. A dichotomous type of any agent is characterized by a positive real number, which we call the *value* of the agent at this type, and a nonempty subset of alternatives, which we call the *acceptable* alternatives. The interpretation is that an agent of dichotomous type gets (the same) utility equal to his value from each alternative in his acceptable set, but gets zero utility on any alternative that

Debasis Mishra: dmishra@isid.ac.in
Souvik Roy: souvik.2004@gmail.com
We are grateful to Sushil Bikhchandani, Dinko Dimitrov, Ron Lavi, Anup Pramanik, Arunava Sen, Tridib Sharma, and seminar audiences at Indian Statistical Institute, Delhi and Delhi School of Economics for valuable suggestions and feedbacks. The feedback of two anonymous referees and a co-editor (Johannes Horner) has been immensely valuable and improved the paper significantly.

DOI: 10.3982/TE1239

is not acceptable. Note that both the value and the set of acceptable alternatives are private information of the agent. This makes such type spaces multidimensional.

We call a type space a *dichotomous* domain if every type belonging to it is a dichotomous type. We characterize the set of implementable allocation rules in dichotomous domains using a condition called *generation monotonicity*. Generation monotonicity is a new (nontrivial) simplification of the *cycle monotonicity* condition of Rochet (1987) in dichotomous domains. Our most striking result comes in a particular class of dichotomous domains. We show that for a large class of dichotomous domains, which we refer to as *rich* dichotomous domains, a significantly weaker condition than generation monotonicity characterizes implementability. We refer to this weaker condition as 2-generation monotonicity and show it to be equivalent to 3-cycle monotonicity. Three-cycle monotonicity is significantly weaker than cycle monotonicity but stronger than 2-cycle monotonicity, a condition used to characterize implementability in convex domains (Bikhchandani et al. 2006, Saks and Yu 2005, Ashlagi et al. 2010). A dichotomous domain is not convex, but is still multidimensional. While most of the earlier results in the literature found domains where 2-cycle monotonicity is necessary and sufficient for implementability, to our knowledge, this paper is the first to identify multidimensional domains where we see that K-cycle monotonicity ($K \neq 2$) is necessary and sufficient for implementation. We show, by way of an example, that 2-cycle monotonicity is not sufficient for implementability in rich dichotomous domains. We demonstrate the usefulness of our characterizations by deriving a revenue maximizing mechanism for the one-sided matching problem where agents have dichotomous preferences over alternatives.

Although dichotomous types seem like a restrictive preference over alternatives, it is natural in many settings. Such preferences are studied in social choice theory and matching theory in models without monetary transfers: Bogomolnaia and Moulin (2004) and Roth et al. (2005) study it in the context of matching; Bogomolnaia et al. (2005) study it in a collective choice problem; Vorsatz (2007, 2008) studies it in the context of a voting model. Allowing for transfers in some of these models is very natural. Dichotomous domains were first studied with monetary transfers and quasilinear utility in Babaioff et al. (2005). We discuss two broad settings with transfers where it is plausible to assume that agents have dichotomous types.

Collective choice. In collective choice problems, agents want to collectively choose an alternative, e.g., joint hiring of a staff member/expert by several departments in a university/firm, joint installation of software for employees in an organization, and choosing a communication or transportation network to build for joint use. In each of these problems, it is plausible to think that agents have dichotomous preferences over alternatives: in the staff hiring example, a department gets a value from a staff member if and only if he has the skills required by the department; in the software installation problem, an employee gets a value from software if and only if it is compatible with his laptop; in the network selection problem, if each agent uses the network to send data from a source node to a destination node, then he gets a value from the network if and only if it connects his source and destination nodes. Allowing for transfers in these problems is natural: in the staff hiring problem, each department contributes to the expert's salary;

in the network building problem, each user pays a price to use the network; in the software installation problem, an organization may charge the employees whose software is compatible and compensate the employees whose software is not compatible.

Private good allocation. In private good allocation problems, each agent receives a different alternative and there is usually some feasibility constraint linking the allocations of all the agents. For example, in scheduling problems, each agent has a task (a journey) that can be completed by a machine (airline). The tasks (journeys) of different agents need to be assigned to different time periods because the machine (airline) has a capacity constraint in each time period. But an agent may not be available in some time periods, and he gets a value if and only if the task (journey) is assigned to a time period when he is available. Related to this example is the general model of matching with transfers in dichotomous domains, for example, in matching firms to job candidates (where transfers are salaries of the candidates), a firm may get a value from a candidate if and only if the candidate has the required skills. Transfers are usually permitted in such job matching problems; see Crawford and Knoer (1981). The single-minded combinatorial auctions domain (Lehmann et al. 2002) is another example of a dichotomous domain. Here, a set of objects is sold to a set of bidders. Each bidder is interested only in a particular subset of objects, called his *favorite bundle*. A bidder gets a value from an alternative (a subset of objects) if and only if it includes his favorite bundle.

Our general characterization using generation monotonicity applies to all these domains. Our specific characterization using 2-generation monotonicity (3-cycle monotonicity) applies to all the above domains except the single-minded combinatorial auction domain.

1.1 *Past literature and our results*

The study of implementable allocation rules in quasilinear utility settings with private values began in the seminal paper of Myerson (1981), who studies Bayes–Nash randomized implementation for the one-dimensional model of the single object auction. For deterministic allocation rules and dominant strategy implementation, Myerson's result can be easily adapted as follows. He defines the notion of monotone allocation rules, which states that given the type profile of other agents, if an agent gets the object at a type, then he must get the object at a type with higher value. Myerson shows that an allocation rule is implementable if and only if it is monotone in this sense; see extensions of this result for various other one-dimensional problems in Archer and Tardos (2001), Archer et al. (2003), Goldberg and Hartline (2005), Aggarwal and Hartline (2006), and Dhangwatnotai et al. (2008).

For a general multidimensional type space model, Rochet (1987) shows that implementability is equivalent to *cycle monotonicity*, which requires that for every agent and for every type profile of other agents, certain *type graphs* should have no cycles of negative length.[1]

[1] This interpretation of cycle monotonicity is due to Gui et al. (2004) and Heydenreich et al. (2009). The cycle monotonicity characterization of implementability is related to the characterization of subgradients of convex functions using cycle monotonicity by Rockafellar (1970).

While the cycle monotonicity characterization is very general, it is not an easy condition to verify or interpret; see extensions and different interpretations in Rahman (2011) and Kos and Messner (2013). Researchers have since tried to identify domains where a simpler condition than cycle monotonicity is necessary and sufficient for (deterministic) implementability. Bikhchandani et al. (2006) show that 2-cycle monotonicity, which requires cycles that have two nodes in the type graph to have nonnegative length, is necessary and sufficient for implementability in a variety of convex domains, including the unrestricted domain and some auction domains. Saks and Yu (2005) generalize this result to show that 2-cycle monotonicity is necessary and sufficient for implementability if the type space of every agent is a convex subset of $\mathbb{R}^{|\mathcal{A}|}$, where $\mathcal{A}$ is the set of alternatives. Ashlagi et al. (2010, supplementary material) extend this result to show that 2-cycle monotonicity is necessary and sufficient for implementability if the closure of type space of every agent is a convex subset of $\mathbb{R}^{|\mathcal{A}|}$. Vohra (2011) has an excellent survey of these results.[2] Note that the 2-cycle monotonicity condition is equivalent to Myerson's monotonicity condition in the single object auction model.[3] Our characterization of implementability in rich dichotomous domains uses 2-generation monotonicity, which is equivalent to 3-cycle monotonicity. Since 3-cycle monotonicity is slightly stronger than 2-cycle monotonicity and since we require 3-cycle monotonicity—not 2-cycle monotonicity—to characterize implementability, our result helps to further delineate the boundaries of multidimensional domains that permit a characterization that is significantly simpler than Rochet's cycle monotonicity.

This paper is not the first paper to study implementation in dichotomous domains. Lehmann et al. (2002) consider the specific dichotomous domain of single-minded combinatorial auctions. Under an additional assumption on allocation rules, Lehmann et al. (2002) show that 2-cycle monotonicity characterizes implementability in these domains. Our results are more general than this in the sense that we characterize implementability in arbitrary dichotomous domains. Further, our main characterization in rich dichotomous domains does not apply to single-minded auction domains since such domains are not rich in our sense.

A paper closely related to our work is Babaioff et al. (2005). Like us, they consider deterministic implementation in dichotomous domains with monetary transfers. The main difference between their characterization and our characterization is that theirs is a characterization of "mechanisms" (allocation rules and payments), while ours is a characterization of "allocation rules" only. Their characterization says that a mechanism is *truthful* if and only if the corresponding allocation rule is *value monotone, encourages*

[2]Vohra (2011) and Heydenreich et al. (2009) discuss an alternate graph theoretic interpretation of cycle monotonicity using *allocation graphs*. Cuff et al. (2012) show that if the type space is a full-dimensional convex product space, then implementability is equivalent to every 2-cycle in the allocation graph having *zero* length. Since an allocation graph is a more complicated concept than the type graph, we do not discuss it in detail in this paper.

[3]There are many papers that characterize different extensions of implementability in convex domains using 2-cycle monotonicity and additional technical conditions: for Bayes–Nash implementation, see Jehiel et al. (1999) and Müller et al. (2007); for randomized implementation, see Archer and Kleinberg (2008); for implementation with general value functions, see Berger et al. (2010) and Carbajal and Ely (2012).

winning, ensures minimal payments, and the *payments are by critical values*.[4] Our view is that our direct characterizations of implementable allocation rules are simpler to state and very different in spirit from the result in Babaioff et al. (2005).

Importantly, our general characterization has many nice implications on specific dichotomous domains, but the characterization in Babaioff et al. (2005) is silent in such domains. Our general characterization using generation monotonicity identifies many specific dichotomous domains where weaker versions of cycle monotonicity are necessary and sufficient for implementability. In rich dichotomous domains, where 2-generation monotonicity characterizes implementability, it implies a *cutoff-based* characterization of implementable allocation rules. This cutoff-based characterization extends the cutoff-based characterization of Myerson (1981) for single object auctions, which states that for every agent and for every type profile of other agents, there is a cutoff value above which this agent gets the object and below which he does not get the object; see also Archer and Tardos (2001) for a generalization of this cutoff-based characterization to general one-dimensional models. Our cutoff-based characterization for the rich dichotomous domains is more involved.

We hope that such simple characterizations lead to identification of optimal mechanisms, mechanisms with fairness properties, and (almost) budget-balanced mechanisms in our model. Further, efficiency is usually computationally difficult in many dichotomous domains; for example, in single-minded combinatorial auction domains (Blumrosen and Nisan 2007). So characterizing the entire class of implementable allocation rules helps us to identify computationally tractable but approximately efficient implementable allocation rules.

We demonstrate the usefulness of our results by deriving the optimal mechanism for a particular setting. We consider the one-sided matching problem where agents have dichotomous preferences. In this problem, a set of objects (say, airline or movie tickets) need to be assigned to a set of agents, where each agent can be assigned at most one object. Each agent finds only a subset of the objects acceptable and derives a value if any of these objects are assigned to him. Such a domain easily satisfies the assumptions of a rich dichotomous domain. Among the class of dominant strategy incentive compatible and individually rational mechanisms, we identify a mechanism that results in maximum expected revenue for the designer in this problem. Our optimal mechanism extends the optimal auction for the single object case in Myerson (1981).

Our derivation of an optimal mechanism for the one-sided matching problem with dichotomous preferences is a contribution to the optimal multidimensional mechanism design literature. The multidimensional optimal mechanism design problem is believed to be a hard problem. There is a long literature to it after Myerson's seminal work on the single object auction; see Rochet and Stole (2003) for a survey. This literature usually considers Bayes–Nash randomized implementation. The usual approach in this literature is to consider specific multidimensional domains (sometimes with relaxed incentive constraints) and then extend Myerson's methodology to such settings; see Armstrong (1996), Blackorby and Szalay (2007), Iyengar and Kumar (2008),

[4]For precise definitions of these terms, we refer the reader to Babaioff et al. (2005).

Pai and Vohra (2010), and Manelli and Vincent (2007). Our optimal mechanism design looks at a different multidimensional domain with deterministic dominant strategy implementation.

2. The single agent model

We consider a single agent model now. Later, we show how we can extend our results to n agents. The interpretation of the single agent model is that the type profile of other agents is fixed and we are looking at the image of an allocation rule where this agent's type is changing.

The single agent is denoted by i. There is a finite set of alternatives $\mathcal{A}$. The type of agent i is a vector in $\mathbb{R}^{|\mathcal{A}|}$. We denote the type of agent i as t_i and denote the value of any alternative $a \in \mathcal{A}$ at type t_i as $t_i(a)$. The set of all possible types of agent i is denoted as $\mathcal{D}_i$ and is referred to as the domain. Agent i has quasilinear utility functions, i.e., if he pays a monetary transfer of p_i and the alternative he receives is $a \in \mathcal{A}$, then his net utility is given by $t_i(a) - p_i$. We also assume private values, so when we consider the model with n agents, the net utility of every agent is completely determined by his own type and his own monetary transfers.

An *allocation rule* f is a mapping $f : \mathcal{D}_i \to \mathcal{A}$. A payment function of agent i is a mapping $p_i : \mathcal{D}_i \to \mathbb{R}$.

Definition 1. An allocation rule f is *implementable* (in dominant strategies) if there exists a payment function p_i such that for every $s_i, t_i \in \mathcal{D}_i$,

$$s_i(f(s_i)) - p_i(s_i) \geq s_i(f(t_i)) - p_i(t_i).$$

In such a case, we say that p_i implements f.

Note that our notion of implementation is of dominant strategies. We discuss a familiar notion of monotonicity for the allocation rules and its relation to implementability.

Definition 2. An allocation rule f is K*-cycle monotone*, where $K \geq 2$ is a positive integer, if for every finite sequence of types $(t_i^1, t_i^2, \dots, t_i^k)$ with $k \leq K$, we have

$$\sum_{j=1}^{k} \left[t_i^j(f(t_i^j)) - t_i^j(f(t_i^{j-1})) \right] \geq 0, \tag{1}$$

where $t_i^0 \equiv t_i^k$. An allocation rule f is *cyclically monotone* if it is K-cycle monotone for all positive integers $K \geq 2$.

Remark. If an allocation rule f is $(K+1)$-cycle monotone, then it is also K-cycle monotone.

In a seminal work, Rochet (1987) shows that an allocation rule is implementable if and only if it is cyclically monotone; also see Rockafellar (1970). The explicit graph theoretic interpretation is due to Gui et al. (2004), who associate a *type graph* with every domain $\mathcal{D}_i$, every set of alternatives $\mathcal{A}$, and every allocation rule $f : \mathcal{D}_i \to \mathcal{A}$. This type graph contains the set of types as the set of nodes and is a complete graph (i.e., a directed edge exists from every node to every other node). The length of the edge from node s_i to t_i is

$$\ell(s_i, t_i) := t_i(f(t_i)) - t_i(f(s_i)).$$

Then it is easy to notice that inequality (1) requires the length of the cycle $(t_i^1, \dots, t_i^k, t_i^1)$ to be nonnegative.

Though mathematically elegant, this characterization of implementability involves verifying the length of cycles that involve an arbitrary number of nodes. When the set of alternatives is finite, as is assumed here, one needs only to verify cycles involving no more than $|\mathcal{A}|$ nodes. The following result is folklore, but, to our knowledge, is not stated explicitly in the literature.

Lemma 1. *An allocation rule f is implementable if and only if it is $|\mathcal{A}|$-cycle monotone, where $\mathcal{A}$ is a finite set of alternatives.*

The proof is given in the Appendix.

The most general result in the literature, due to Ashlagi et al. (2010, supplementary material), shows that if the closure of a domain is convex, then 2-cycle monotonicity is sufficient for implementation. This is a significant improvement over Lemma 1.

3. Implementation in dichotomous domains

We now introduce the domain we study in this paper. We call this domain the *dichotomous domain*.

Definition 3. A type $t_i \in \mathbb{R}^{|\mathcal{A}|}$ is called a *dichotomous type* if there exists a positive real number $v(t_i) \in \mathbb{R}_{++}$ and a nonempty subset of alternatives $A(t_i) \subseteq \mathcal{A}$ such that $t_i(a) = v(t_i)$ if $a \in A(t_i)$ and $t_i(a) = 0$ if $a \notin A(t_i)$.

The alternatives in $A(t_i)$ are called *acceptable* alternatives of agent i at t_i and the positive real number $v(t_i)$ is called the *value* of agent i at t_i. The set $A(t_i)$ is referred to as the acceptable set of agent i at t_i.

We refer to the tuple of acceptable set and value as the *type* of the agent. A domain $\mathcal{D}_i \subseteq \mathbb{R}_+^{|\mathcal{A}|}$ is called a *dichotomous domain* if every $t_i \in \mathcal{D}_i$ is a dichotomous type. For simplicity, we sometimes write the dichotomous type t_i as $(v(t_i), A(t_i))$.

As an example, consider $\mathcal{A} = \{a, b, c, d\}$. The possible acceptable sets of agent i are all nonempty subsets of $\mathcal{A}$. For instance, if $\{a, d\}$ is the acceptable set with value 5, then the type of agent i is $(5, \{a, d\})$. An allocation rule selects one of the alternatives in $\mathcal{A}$, and, in this case, agent i gets a value of 5 if a or d is selected and gets zero value otherwise.

Note that there may be restrictions in a dichotomous domain. For example, a particular alternative in $\mathcal{A}$ may never be acceptable to the agent: such an alternative always has value zero and is referred to as a *worthless* alternative. For example, in the single-minded combinatorial auction setting, one alternative is to give no object to the agent. Such an alternative always gives zero value to the agent and is worthless.

Another restriction can be that if a particular alternative is in the acceptable set, then some other alternative also has to be in the acceptable set. Later, we give specific domains where such restrictions are natural. However, our general result is not influenced by any such restrictions.

The dichotomous domain is not convex as the following example illustrates.

EXAMPLE 1. Let $\mathcal{A} = \{a, b, c\}$. Consider a type where the acceptable set is $\{a, b\}$ and the value is 2, $t_i = (2, 2, 0)$, and another type where the acceptable set is $\{a, c\}$ and the value is 3, $s_i = (3, 0, 3)$. Now $(s_i + t_i)/2 = (2.5, 1, 1.5)$, which is not a dichotomous type. $\diamondsuit$

As a result, the earlier results in the literature that 2-cycle monotonicity is equivalent to implementability do not apply in dichotomous domains.

3.1 *Generation monotonicity*

We examine the implication of implementability in dichotomous domains. Unless stated otherwise, $\mathcal{D}_i$ is a dichotomous domain in this section. The outcome of an allocation rule at a dichotomous type is easy to describe: an agent either gets an alternative in his acceptable set or gets something outside his acceptable set. For every alternative $a \in \mathcal{A}$ and for every dichotomous type t_i, we define the indicator function $\delta(a, t_i) \in \{0, 1\}$, where $\delta(a, t_i) = 1$ implies that $a \in A(t_i)$ and $\delta(a, t_i) = 0$ implies that $a \notin A(t_i)$. Note that in the type graph of a dichotomous domain, the length of the edge from type s_i to type t_i can be written as

$$\ell(s_i, t_i) = t_i(f(t_i)) - t_i(f(s_i)) = v(t_i)\big[\delta(f(t_i), t_i) - \delta(f(s_i), t_i)\big].$$

We now describe a new monotonicity property in dichotomous domains and show it to be equivalent to implementability. For this, we need some notation. Given an allocation rule f, a type t_i is *satisfied* by s_i if $\delta(f(s_i), t_i) = 1$. If $\delta(f(t_i), t_i) = 1$, we say that t_i is satisfied (by itself). If t_i is not satisfied, then we say it is unsatisfied.

As an example, consider $\mathcal{A} = \{a, b, c, d\}$ and an allocation rule f. Let $s_i = (2, \{a, b\})$ and $f(s_i) = c$. In this case, s_i is unsatisfied. Let $t_i = (3, \{b, d\})$ and $f(t_i) = b$. In this case, t_i is satisfied. Further, s_i is satisfied by t_i since $\delta(f(t_i), s_i) = 1$.

The idea of satisfaction comes from analyzing lengths of edges in the type graph. If $\ell(s_i, t_i) < 0$ for some s_i, t_i, then it must be that t_i is unsatisfied and it is satisfied by s_i.

We now define the notion of *generations* of unsatisfied types. For an allocation rule f, define the first generation types of an unsatisfied type $t_i \in \mathcal{D}_i$ as

$$G_1^f(t_i) = \big\{s_i \in \mathcal{D}_i : \delta(f(s_i), t_i) = 1\big\}.$$

So $G_1^f(t_i)$ contains all the types that satisfy t_i. Of course, it does not contain t_i since we consider generations of unsatisfied types only. Also, $G_1^f(t_i)$ may be empty.

Having defined the kth-generation types of the unsatisfied type t_i, we define the $(k+1)$st generation types of t_i as

$$G_{k+1}^f(t_i) = \left\{ s_i \in \mathcal{D}_i \setminus \bigcup_{j=1}^{k} G_j^f(t_i) : \delta(f(s_i), \bar{t}_i) = 1 \text{ for some } \bar{t}_i \in G_k^f(t_i) \right\}.$$

So $G_{k+1}^f(t_i)$ contains all the types that satisfy a kth-generation type of t_i. Note that for every unsatisfied type t_i and every other type s_i, either s_i is not in any generation of t_i or s_i belongs to a unique generation of t_i. It is possible that for an unsatisfied type $t_i \in \mathcal{D}_i$, $G_k^f(t_i) = \varnothing$ for some k. Further, if t_i is unsatisfied and $s_i \in G_k^f(t_i)$ for some generation k, then there is no restriction that s_i itself is satisfied or not. We show later that implementability requires s_i to be satisfied. Finally, whenever $G_k^f(t_i) = \varnothing$, we have $G_{k+1}^f(t_i) = \varnothing$.

We give an example to clarify the concept of generations.

Example 2. Let $\mathcal{A} = \{a, b, c\}$. Suppose t_i is a dichotomous type with $v(t_i) = 2$ and $A(t_i) = \{a\}$. Consider an allocation rule f such that $f(t_i) = b$. Hence, t_i is not satisfied. Now consider a type $\bar{t}_i$ with $v(\bar{t}_i) = 3$ and $A(\bar{t}_i) = \{a, b\}$, and let $f(\bar{t}_i) = a$. Hence, $f(\bar{t}_i) \in A(t_i)$ and this implies that $\bar{t}_i \in G_1^f(t_i)$. Now consider another type $\hat{t}_i$ such that $v(\hat{t}_i) = 1$ and $A(\hat{t}_i) = \{b\}$, and let $f(\hat{t}_i) = b$. Then $\hat{t}_i$ satisfies $\bar{t}_i$ but it does not satisfy t_i. Since $\bar{t}_i$ satisfies t_i, we have $\hat{t}_i \in G_2^f(t_i)$. $\Diamond$

We show that the number of generations of an unsatisfied type for any allocation rule is finite.

Lemma 2. *Suppose $f : \mathcal{D}_i \to \mathcal{A}$ is an allocation rule. For all $t_i \in \mathcal{D}_i$ such that $f(t_i) \notin A(t_i)$, if $G_k^f(t_i) \neq \varnothing$, then $k \leq |\mathcal{A}|$.*

Proof. Fix any $f : \mathcal{D}_i \to \mathcal{A}$, and consider $t_i \in \mathcal{D}_i$ such that $f(t_i) \notin A(t_i)$. Suppose $G_k^f(t_i) \neq \varnothing$ and assume, to the contrary, $k > |\mathcal{A}|$. Then for each positive integer $j \leq k$, we pick some $t_i^j \in G_j^f(t_i)$. Now consider the set of types $\{t_i^1, \ldots, t_i^k\}$. Since $|\mathcal{A}| < k$, there must exist at least two types, say t_i^j and $t_i^{j'}$ with $j, j' \in \{1, \ldots, k\}$, such that $f(t_i^j) = f(t_i^{j'})$. Then it must be that t_i^j and $t_i^{j'}$ belong to the same generation of t_i. This is a contradiction. $\square$

For an allocation rule f, define the generation number of an unsatisfied type t_i in f as the largest positive integer $\gamma^f(t_i)$ such that $G_{\gamma^f(t_i)}^f(t_i) \neq \varnothing$. By Lemma 2, it is well defined. We now define a monotonicity property using generations of types and show its connection to cycle monotonicity.

Definition 4. An allocation rule f is *K-generation monotone*, where K is a positive integer, if for every unsatisfied type $t_i \in \mathcal{D}_i$ and for every positive integer $k \leq K$, the following statements hold for all $s_i \in G_k^f(t_i)$.

- Generation self-satisfaction (GSS): Type s_i is satisfied.
- Monotonicity (MON): We have $v(s_i) \geq v(t_i)$.

An allocation rule f is *generation monotone* if it is K-generation monotone for all positive integers K.

We strengthen the notion of generation monotonicity below.

Definition 5. An allocation rule f is *strong K-generation monotone*, where K is a positive integer, if it is K-generation monotone, and for every unsatisfied type $t_i \in \mathcal{D}_i$ and for every positive integer $k \leq K$, the following statement holds for all $s_i \in G^f_k(t_i)$.

- No rebirth (NR): Type t_i does not satisfy s_i.

An allocation rule f is *strong generation monotone* if it is strong K-generation monotone for all positive integers K.

Consider the allocation rule in Example 2. This allocation rule fails 2-generation monotonicity. To see this, note that $\hat{t}_i \in G^f_2(t_i)$ but $v(\hat{t}_i) = 1 < v(t_i) = 2$, violating MON. It also fails strong 1-generation monotonicity because $\bar{t}_i \in G^f_1(t_i)$ but t_i satisfies $\bar{t}_i$, violating NR.

Strong generation monotonicity and generation monotonicity are related in an obvious way.

Lemma 3. *Suppose f is $(K+1)$-generation monotone. Then it is strong K-generation monotone.*

Proof. Suppose f is $(K+1)$-generation monotone. Assume, to the contrary, that f is not strong K-generation monotone. Then, for some t_i such that $f(t_i) \notin A(t_i)$ and for some $s_i \in G^f_k(t_i)$, where $k \leq K$, we have that t_i satisfies s_i (a violation of NR). Then $t_i \in G^f_{k+1}(t_i)$. Since f is $(k+1)$-generation monotone, t_i satisfies itself (GSS). This is a contradiction. □

Lemmas 2 and 3 immediately establish the following corollary.

Corollary 1. *An allocation rule is strong generation monotone if and only if it is generation monotone.*

To understand why generation monotonicity may be linked to implementability (cycle monotonicity), consider 2-cycle monotonicity. Consider two types t_i and s_i. The length of the 2-cycle between s_i and t_i is

$$v(t_i)\big[\delta(f(t_i), t_i) - \delta(f(s_i), t_i)\big] + v(s_i)\big[\delta(f(s_i), s_i) - \delta(f(t_i), s_i)\big].$$

For this cycle to have nonnegative length, we need to ensure that when one of the edges has negative length, the other edge must have sufficiently large positive length. Suppose the edge length $\ell(s_i, t_i) < 0$. Then it must be that $\delta(f(t_i), t_i) = 0$ and $\delta(f(s_i), t_i) = 1$, i.e., $s_i \in G_1^f(t_i)$. The length of this edge is $-v(t_i)$. For the 2-cycle to have nonnegative length, we must have $\delta(f(s_i), s_i) = 1$ (GSS), $\delta(f(t_i), s_i) = 0$ (NR), and $v(s_i) \geq v(t_i)$ (MON). This intuition carries forward to higher generations. The following proposition establishes the exact connection between generation monotonicity and cycle monotonicity.

Proposition 1. *For any positive integer $K \geq 2$, an allocation rule is K-cycle monotone if and only if it is strong $(K-1)$-generation monotone.*

The long proof is given in the Appendix.

We now give a characterization of implementable allocation rules using generation monotonicity. For this, we define certain notions. The *generation number* of an allocation rule $f : \mathcal{D}_i \to \mathcal{A}$ is a positive number defined as follows. If every $t_i \in \mathcal{D}_i$ is satisfied or every unsatisfied $t_i \in \mathcal{D}_i$ is not satisfied by any other type (i.e., $G_1^f(t_i) = \varnothing$ for all t_i), then we let $\gamma^f = 1$; else,

$$\gamma^f = \max_{t_i \in \mathcal{D}_i : f(t_i) \notin A(t_i)} \gamma^f(t_i).$$

By Lemma 2, the value of $\gamma^f \leq |\mathcal{A}|$. We are now ready to state our main characterization.

Theorem 1. *Suppose $f : \mathcal{D}_i \to \mathcal{A}$ is an allocation rule with generation number γ^f. Then the statements*

(i) f is implementable

(ii) f is $(\gamma^f + 1)$-cycle monotone

(iii) f is γ^f-generation monotone

are equivalent.

Proof. (i) $\Rightarrow$ (ii) follows from the fact that implementability implies cycle monotonicity. For (ii) $\Rightarrow$ (iii), note that if f is $(\gamma^f + 1)$-cycle monotone, by Proposition 1, it is strong γ^f-generation monotone and, hence, γ^f-generation monotone. Finally, for (iii) $\Rightarrow$ (i), suppose f is γ^f-generation monotone. Then, by definition of γ^f, f is generation monotone (this follows from the observation that for any positive integer $k > \gamma^f$ and for any t_i such that $f(t_i) \notin A(t_i)$, we have $G_k^f(t_i) = \varnothing$). In that case, by Lemma 3, f is strong generation monotone. By Proposition 1, f satisfies cycle monotonicity. Hence, f is implementable. □

Theorem 1 serves as the building block for our main result in Section 4. In the working paper version of this paper (Mishra and Roy 2012), we identify specific dichotomous domains where we can find the generation number and, using Theorem 1, we get immediate characterizations in these domains.

4. Rich dichotomous domain

We now move beyond the general characterization in Theorem 1. We impose an additional *richness* assumption on the domain and use Theorem 1 to get a simpler characterization of implementability in these domains. Our richness condition rules out some restrictions that may arise in dichotomous domains.

Definition 6. A dichotomous domain $\mathcal{D}_i$ is *rich* if

(a) the set of possible values of a dichotomous type is an interval $V = (0, \beta)$,[5] where $\beta \in \mathbb{R}_{++} \cup \{\infty\}$

(b) for every alternative $a \in \mathcal{A}$ that is not worthless [6] and every possible value $x \in V$, there is a dichotmous type t_i such that $v(t_i) = x$ and $A(t_i) = \{a\}$.

Condition (a) is plausible in almost all dichotomous domains. However, condition (b) may not be satisfied in some domains. In particular, it is clearly violated in the single-minded combinatorial auction domain. Recall that, in the single-minded domain, an auctioneer is selling a set of m objects and the bidder is interested only in a subset of objects, called his favorite bundle. The set of alternatives in this problem is the set of all subsets of objects. However, if a single-minded bidder has a particular subset of objects S in his acceptable set, then he must have *every* superset of S in his acceptable set. This is a particular restriction on this dichotomous domain, which rules out richness.

There are many interesting domains where condition (b) holds. For example, it holds in all the collective choice problems we discussed in Section 1. It also holds in some private good allocation problems that we discussed in Section 1, e.g., in the scheduling problem and in the matching problem. Thus, it covers a wide variety of dichotomous domains.

The main result of this section gives a characterization of implementable allocation rules in rich dichotomous domains.

Theorem 2. *For any allocation rule $f : \mathcal{D}_i \to \mathcal{A}$, where $\mathcal{D}_i$ is a rich dichotomous domain, the statements*

(i) f is implementable

(ii) f is 3-cycle monotone

(iii) f is 2-generation monotone

are equivalent.

The proof of Theorem 2 relies on a particular type of payment function that we construct. For this, we define a function $\kappa_i^f : \mathcal{A} \to \mathbb{R}_+$ for an allocation rule f as follows.

[5]Our results are true even if we consider intervals of the form $(0, \beta]$.

[6]As defined earlier, an alternative is worthless if it can *never* be in the acceptable set at any dichotomous type.

If $f(s_i) \neq a$ for all s_i with $A(s_i) = \{a\}$, then we let $\kappa_i^f(a) = 0$. Also, if a is a worthless alternative, then $\kappa_i^f(a) = 0$. Otherwise, for every other $a \in \mathcal{A}$, let

$$\kappa_i^f(a) = \inf\{v(s_i) \in V : s_i \in \mathcal{D}_i, f(s_i) = a, A(s_i) = \{a\}\}. \tag{2}$$

In words, $\kappa_i^f(a)$ is the minimum value at which any dichotomous type containing only a in the acceptable set is satisfied. Because of our richness assumption, for all $a \in \mathcal{A}$, $\kappa_i^f(a)$ is well defined. Note that $\kappa_i^f(a) \geq 0$ for all $a \in \mathcal{A}$.

Now we define a payment function p_i^* as follows. Given an allocation rule f, for every $s_i \in \mathcal{D}_i$, define

$$p_i^*(s_i) = \kappa_i^f(f(s_i))\delta(f(s_i), s_i).$$

Note that an agent pays zero if he is not satisfied at a type.

Proposition 2. *Suppose $f : \mathcal{D}_i \to \mathcal{A}$ is 2-generation monotone, where $\mathcal{D}_i$ is a rich dichotomous domain. Then p_i^* implements f.*

Proof. To show that p_i^* implements f, we consider two types t_i and s_i in $\mathcal{D}_i$. We show that

$$v(t_i)\delta(f(t_i), t_i) - p_i^*(t_i) \geq v(t_i)\delta(f(s_i), t_i) - p_i^*(s_i)$$

or

$$\left[v(t_i) - \kappa_i^f(f(t_i))\right]\delta(f(t_i), t_i) \geq v(t_i)\delta(f(s_i), t_i) - \kappa_i^f(f(s_i))\delta(f(s_i), s_i).$$

The left-hand side is referred to as the *payoff from truth* and the right-hand side is referred to as the *payoff from lie*.

We consider various cases.

Case 1. Suppose $\delta(f(t_i), t_i) = 0$ and $\delta(f(s_i), t_i) = 0$. Then the payoff from truth is zero and the payoff from lie is nonpositive. Hence, we are done.

Case 2. Suppose $\delta(f(t_i), t_i) = 0$ and $\delta(f(s_i), t_i) = 1$. Since $s_i \in G_1^f(t_i)$, by GSS, $\delta(f(s_i), s_i) = 1$. Hence, the payoff from truth is 0 and the payoff from lie is $v(t_i) - \kappa_i^f(f(s_i))$. Assume, to the contrary, that $v(t_i) > \kappa_i^f(f(s_i))$. Consider a type $\bar{s}_i \in \mathcal{D}_i$ such that $v(\bar{s}_i) = \kappa_i^f(f(s_i)) + \epsilon < v(t_i)$, where $\epsilon > 0$ but arbitrarily close to zero and $A(\bar{s}_i) = \{f(s_i)\}$. By definition of κ_i^f, there is some type $\hat{s}_i$ with $A(\hat{s}_i) = \{f(s_i)\}$ and $v(\hat{s}_i)$ arbitrarily close to $\kappa_i^f(f(s_i))$ such that $f(\hat{s}_i) = f(s_i)$. Then 1-generation monotonicity implies that $f(\bar{s}_i) = f(\hat{s}_i) = f(s_i)$. Since $f(s_i) \in A(t_i)$, $\bar{s}_i$ satisfies t_i. But $\delta(f(t_i), t_i) = 0$ implies that $\bar{s}_i \in G_1^f(t_i)$. By MON, $v(\bar{s}_i) \geq v(t_i)$. This is a contradiction. This shows that $v(t_i) - \kappa_i^f(f(s_i)) \leq 0$ and, hence, we are done.

Case 3. Suppose $\delta(f(t_i), t_i) = 1$ and $\delta(f(s_i), t_i) = 0$. In such a case, the payoff from lie is nonpositive. The payoff from truth is $v(t_i) - \kappa_i^f(f(t_i))$, which we show to be nonnegative. Assume, to the contrary $v(t_i) < \kappa_i^f(f(t_i))$. Consider a type $\bar{t}_i \in \mathcal{D}_i$ such that $v(\bar{t}_i) = v(t_i) + \epsilon$, where $\epsilon > 0$ and arbitrarily close to zero, and $A(\bar{t}_i) = \{f(t_i)\}$. If $f(\bar{t}_i) \neq f(t_i)$, then $t_i \in G_1^f(\bar{t}_i)$. But $v(\bar{t}_i) > v(t_i)$ violates MON. Hence, $f(\bar{t}_i) = f(t_i)$. By definition of κ_i^f, $v(\bar{t}_i) \geq \kappa_i^f(f(t_i))$. This is a contradiction since ϵ is sufficiently close to zero.

Case 4. Suppose $\delta(f(t_i), t_i) = 1$ and $\delta(f(s_i), t_i) = 1$. We consider two subcases.

(a) Suppose $\delta(f(s_i), s_i) = 0$. Then the payoff from truth is $v(t_i) - \kappa_i^f(f(t_i))$ and the payoff from lie is $v(t_i)$. We show that $\kappa_i^f(f(t_i)) = 0$. Assume, to the contrary, that $\kappa_i^f(f(t_i)) > \epsilon > 0$, where ϵ is arbitrarily close to zero. Consider another type $\bar{t}_i \in \mathcal{D}_i$ such that $v(\bar{t}_i) = \kappa_i^f(f(t_i)) - \epsilon$ and $A(\bar{t}_i) = \{f(t_i)\}$. By definition of κ_i^f, $f(\bar{t}_i) \neq f(t_i)$. Hence, $t_i \in G_1^f(\bar{t}_i)$ and $s_i \in G_2^f(\bar{t}_i)$ (since $f(s_i) \in A(t_i)$). By GSS, $\delta(f(s_i), s_i) = 1$. This is a contradiction. Hence, $\kappa_i^f(f(t_i)) = 0$ and we are done.

(b) Suppose $\delta(f(s_i), s_i) = 1$. Then the payoff from truth is $v(t_i) - \kappa_i^f(f(t_i))$ and the payoff from lie is $v(t_i) - \kappa_i^f(f(s_i))$. We show that $\kappa_i^f(f(t_i)) \leq \kappa_i^f(f(s_i))$ and we are done. If $f(t_i) = f(s_i)$, then obviously we are done. Else, assume, to the contrary, $\kappa_i^f(f(t_i)) > \kappa_i^f(f(s_i))$. Suppose $\kappa_i^f(f(t_i)) - \kappa_i^f(f(s_i)) = \epsilon > 0$. Then we consider two types $\bar{t}_i$ and $\bar{s}_i$ as follows:

$$\begin{aligned}
v(\bar{t}_i) &= \kappa_i^f(f(t_i)) - \tfrac{1}{3}\epsilon \\
v(\bar{s}_i) &= \kappa_i^f(f(s_i)) + \tfrac{1}{3}\epsilon \\
A(\bar{t}_i) &= \{f(t_i)\} \\
A(\bar{s}_i) &= \{f(s_i)\}.
\end{aligned}$$

By definition of κ_i^f, $f(\bar{t}_i) \neq f(t_i)$ and $f(\bar{s}_i) = f(s_i)$. So t_i satisfies $\bar{t}_i$ and $\bar{s}_i$ satisfies t_i (since $f(s_i) \in A(t_i)$). Hence, $\bar{s}_i \in G_2^f(\bar{t}_i)$. By MON, $v(\bar{s}_i) \geq v(\bar{t}_i)$. This is a contradiction. □

The proof of Theorem 2 is now immediate.

Proof of Theorem 2. Implementability implies 3-cycle monotonicity. Proposition 1 shows that 3-cycle monotonicity implies strong 2-generation monotonicity, which implies 2-generation monotonicity. Proposition 2 shows that 2-generation monotonicity implies implementability. □

4.1 *Two-cycle monotonicity is not sufficient*

In this section, we give an example of an allocation rule in the rich dichotomous domain that satisfies 2-cycle monotonicity but is not implementable. Let $\mathcal{A} = \{a, b, c\}$. An allocation rule f is shown in Figure 1, where all possible acceptable sets are depicted along the top. The allocation rule f has a *cutoff* for each acceptable set. For any acceptable

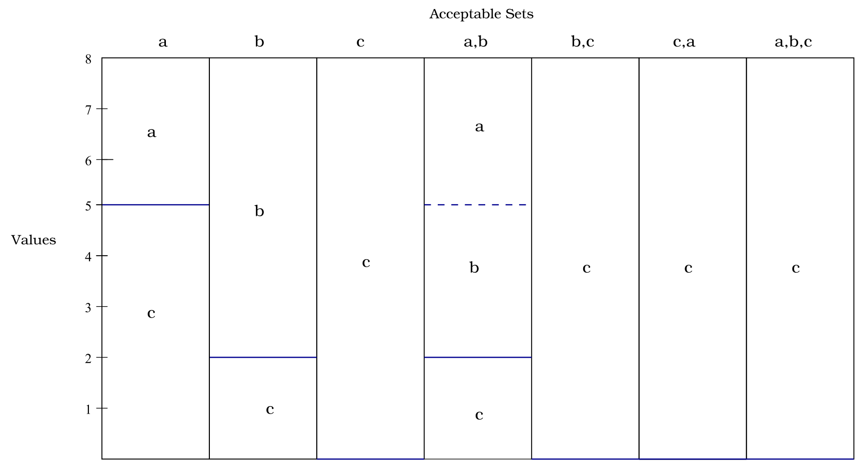

FIGURE 1. A 2-cycle monotone allocation rule that is not 3-cycle monotone.

set, a cutoff specifies a value below which the type is not satisfied and above which the type is satisfied. For example, in Figure 1, the cutoff for acceptable set $\{a\}$ is 5, for $\{b\}$ is 2, for $\{b, c\}$ is zero, and so on. In Figure 1, these cutoffs are indicated by horizontal dark lines that correspond to each acceptable set. The outcomes of the allocation rule below and above these cutoffs are shown in Figure 1. The dashed line indicates the boundary where outcomes change (for a given acceptable set).

One can verify that f is 1-generation monotone. It is also strong 1-generation monotone, i.e., it satisfies NR. To see this, note that the types that are not satisfied in f have acceptable set $\{a\}$ or $\{b\}$ or $\{a, b\}$, and the outcomes at these types are c. Hence, NR is satisfied. By Proposition 1, f is 2-cycle monotone.

But f is not 2-generation monotone and, hence, is not implementable. To see this, consider a type t_i^0 such that $v(t_i^0) = 5 - \epsilon$, where $\epsilon \in (0, 3/2)$, and $A(t_i^0) = \{a\}$. By definition, $f(t_i^0) = c$. Now, consider a type t_i^1 such that $v(t_i^1) = 5 + \epsilon$ and $A(t_i^1) = \{a, b\}$. By definition, $f(t_i^1) = a$. Hence, $t_i^1 \in G_1^f(t_i^0)$. Finally, consider a type t_i^2 such that $v(t_i^2) = 5 - 2\epsilon$ and $A(t_i^2) = \{a, b\}$. By definition, $f(t_i^2) = b$. Hence, $t_i^2 \in G_2^f(t_i^0)$. By 2-generation monotonicity, we must have $v(t_i^2) \geq v(t_i^0)$. But this is not true. Indeed, the 3-cycle involving the nodes $(t_i^0, t_i^1, t_i^2, t_i^0)$ has a length of $-\epsilon < 0$.

In Section 4.3, we revisit this example and give some intuition on why 2-cycle monotonicity is not sufficient but 3-cycle monotonicity is equivalent to implementability in rich dichotomous domains.

4.2 *A characterization using cutoffs*

A remarkable feature of Myerson's monotonicity characterization in the setting of single object auctions is that it implies a simpler characterization using *cutoffs*. In particular, it says that if an allocation rule (which is deterministic) is implementable, then there must exist a cutoff value for the agent such that below this cutoff value, the agent does not get the object and above this value, he does. The aim of this section is to give such a cutoff-based characterization in rich dichotomous domains. A cutoff-based characterization is simple to understand.

First, we define the notion of cutoffs in rich dichotomous domains. It is similar to the κ_i^f that we defined earlier.

Definition 7. A *cutoff* is a mapping $\kappa_i : \mathcal{A} \to \mathbb{R}_+ \cup \{\infty\}$ such that $\kappa_i(a) = 0$ for all $a \in \mathcal{A}$ that are worthless.

Note that there may be alternatives that are *not* worthless and still have zero cutoff. If $\kappa_i(a) = 0$, then a is called a fulfilling alternative of cutoff κ_i. A cutoff κ_i is a *feasible cutoff* if there is some alternative $a \in \mathcal{A}$ that is fulfilling. Feasibility is trivially satisfied if there is a worthless alternative.

Given cutoffs on each alternative, we can define cutoffs on any acceptable set (i.e., any nonempty subset of alternatives). We allow for the fact that not every subset of alternatives may be an acceptable set in certain rich dichotomous domains. Indeed, our richness assumption requires only that singleton alternatives (which are not worthless) can be acceptable sets. Define the set of subsets of alternatives that can be acceptable sets as

$$\Sigma := \{S \subseteq \mathcal{A} : S = A(t_i) \text{ for some } t_i \in \mathcal{D}_i\}.$$

Note that if $S \in \Sigma$, then S does not contain any worthless alternative. By our richness assumption, if $S \in \Sigma$, then for *all* possible values x, $(x, S) \in \mathcal{D}_i$. Also, if a is not a worthless alternative, then richness implies that $\{a\} \in \Sigma$.

Now fix a cutoff mapping κ_i. For any acceptable set $S \in \Sigma$, define the cutoff induced by κ_i on S as

$$C^{\kappa_i}(S) = \min_{a \in S} \kappa_i(a). \tag{3}$$

So the cutoff for an acceptable set S is the minimum over cutoffs of the alternatives in S. Intuitively, cutoffs are like the prices of acceptable sets: prices for individual alternatives define prices for acceptable sets.

Example 3. Let $\mathcal{A} = \{a, b, c\}$ and assume that there are no worthless alternatives. Consider κ_i such that $\kappa_i(a) = 5$, $\kappa_i(b) = 2$, and $\kappa_i(c) = 0$. By (3), $C^{\kappa_i}(\{a\}) = 5$, $C^{\kappa_i}(\{b\}) = C^{\kappa_i}(\{a, b\}) = 2$, and $C^{\kappa_i}(S) = 0$ for all $S \notin \{\{a\}, \{b\}, \{a, b\}\}$. ◊

For any acceptable set $S \in \Sigma$, let

$$W^{\kappa_i}(S) := \{a \in S : C^{\kappa_i}(S) = \kappa_i(a)\}.$$

The set $W^{\kappa_i}(S)$ contains all the alternatives in S that have the same cutoff as S itself. Note that $W^{\kappa_i}(S) \neq \varnothing$. In Example 3, $W^{\kappa_i}(\{a, b\}) = \{b\}$ and $W^{\kappa_i}(\{a, b, c\}) = \{c\}$. Using the price interpretation for cutoffs, the W^{κ_i} set identifies the alternatives in the acceptable set that are the cheapest.

Further, for every $S \in \Sigma$, let

$$L^{\kappa_i}(S) = \{a \notin S : \kappa_i(a) = 0\}.$$

The set $L^{\kappa_i}(S)$ contains all the alternatives outside S that are fulfilling. This set can be potentially empty. For instance, in Example 3, $L^{\kappa_i}(\{b, c\}) = \varnothing$ but $L^{\kappa_i}(\{a\}) = \{c\}$. Intuitively, using the price interpretation for cutoffs, the L^{κ_i} set identifies the zero price alternatives outside the acceptable set. Note that by the definition of feasible cutoff κ_i, for any acceptable set $S \in \Sigma$, if $L^{\kappa_i}(S) = \varnothing$, then $C^{\kappa_i}(S) = 0$, and if $C^{\kappa_i}(S) > 0$, then there is some fulfilling alternative $a \notin S$ such that $\kappa_i(a) = 0$, and this implies that $L^{\kappa_i}(S) \neq \varnothing$.

Now we are ready to formally define a cutoff-based rule, generalizing the idea of a cutoff-based rule in single object auction setting.

Definition 8. An allocation rule f is *cutoff-based* if there exists a feasible cutoff κ_i such that at every dichotomous type $t_i \equiv (A(t_i), v(t_i))$,

(i) if $v(t_i) > C^{\kappa_i}(A(t_i))$, then $f(t_i) \in A(t_i)$, and if $v(t_i) < C^{\kappa_i}(A(t_i))$, then $f(t_i) \notin A(t_i)$

(ii) if $f(t_i) \in A(t_i)$, then $f(t_i) \in W^{\kappa_i}(A(t_i))$, and if $f(t_i) \notin A(t_i)$, then $f(t_i) \in L^{\kappa_i}(A(t_i))$.

In summary, a cutoff-based allocation rule has the following features.

- It specifies cutoffs for each alternative.
- The cutoff for any acceptable set is just the minimum of cutoffs of all the alternatives in that acceptable set (3).
- For any type, if the value is above the cutoff for that acceptable set, then the type is satisfied and an alternative in the W^{κ_i} set is chosen.
- For any type, if the value is below the cutoff for that acceptable set, then the type is unsatisfied and an alternative in the L^{κ_i} set is chosen.

Note how this generalizes the idea of a cutoff-based allocation rule in the single object auction model. This leads us to the main result of this section.

Theorem 3. *Suppose $\mathcal{D}_i$ is a rich dichotomous domain. An allocation rule $f : \mathcal{D}_i \to \mathcal{A}$ is implementable if and only if it is cutoff-based.*

The proof exploits the characterization in Theorem 2. We prove a series of claims that show the implications of 1-generation monotonicity and 2-generation monotonicity. These small steps lead to the characterization of the cutoff-based rule.

Proof of Theorem 3. Let $f : \mathcal{D}_i \to \mathcal{A}$ be an allocation rule, where $\mathcal{D}_i$ is a rich dichotomous domain. Suppose f is implementable. By Theorem 2, f is 2-generation monotone. Then we can define the cutoffs as follows. For every $S \in \Sigma$, let $C_i(S) = \infty$ if for all $t_i \in \mathcal{D}_i$ with $A(t_i) = S$, we have $f(t_i) \notin S$. Else, define

$$C_i(S) = \inf\{v(t_i) : t_i \in \mathcal{D}_i, A(t_i) = S, f(t_i) \in S\}.$$

Since the domain $\mathcal{D}_i$ is rich, for every $t_i \in \mathcal{D}_i$, we have that $v(t_i) \in V = (0, \beta)$ and this ensures that $C_i(S) \geq 0$. Now we make a series of claims.

Claim 1. *If f is 1-generation monotone, then for every $t_i \in \mathcal{D}_i$, $f(t_i) \in A(t_i)$ implies that $v(t_i) \geq C_i(A(t_i))$ and $f(t_i) \notin A(t_i)$ implies that $v(t_i) \leq C_i(A(t_i))$.*

Proof. The first part follows from the definition of C_i. For the second part, suppose that $f(t_i) \notin A(t_i)$ and $v(t_i) > C_i(A(t_i))$. By definition of $C_i(A(t_i))$, there is some type s_i such that $v(s_i)$ is arbitrarily close to $C_i(A(t_i))$ and $A(s_i) = A(t_i)$ such that $f(s_i) \in A(t_i)$. Hence, $s_i \in G_1^f(t_i)$. By 1-generation monotonicity, $v(s_i) \geq v(t_i)$, which is a contradiction since $v(s_i)$ is arbitrarily close $C_i(A(t_i))$. $\lhd$

Claim 2. *If f is 1-generation monotone, then for every $S \in \Sigma$,*

$$C_i(S) = \min_{a \in S} C_i(\{a\}).$$

Proof. Consider any $S \in \Sigma$ and let $\min_{a \in S} C_i(\{a\}) = C_i(\{b\})$. Assume, to the contrary, that $C_i(S) < C_i(\{b\})$. Then consider the type t_i, where $A(t_i) = S$ and $v(t_i) = C_i(S) + \epsilon < C_i(\{b\})$ (such an $\epsilon > 0$ clearly exists). By definition, $f(t_i) \in S$. Let $f(t_i) = c$. Then $C_i(\{c\}) > v(t_i)$ implies that there is some type s_i with $A(s_i) = \{c\}$ and $v(s_i) = C_i(\{c\}) - \epsilon' > v(t_i)$ such that $f(s_i) \neq c$. Hence, t_i satisfies s_i and $t_i \in G_1^f(s_i)$. But 1-generation monotonicity implies that $v(t_i) \geq v(s_i)$, which is a contradiction.

Hence, $C_i(S) \geq C_i(\{b\})$. Assume, to the contrary, that $C_i(S) > C_i(\{b\})$. Consider two types s_i and t_i such that $A(s_i) = S$ and $A(t_i) = \{b\}$ but $v(s_i) = C_i(S) - \epsilon > v(t_i) = C_i(\{b\}) + \epsilon'$ (clearly, such $\epsilon, \epsilon' > 0$ exist). By definition, $f(s_i) \notin S$ and $f(t_i) = b$. This implies that $t_i \in G_1^f(s_i)$. But 1-generation monotonicity implies that $v(t_i) \geq v(s_i)$. This is a contradiction. $\lhd$

Using these claims, we can now define the following well defined cutoff rule. For every $a \in \mathcal{A}$, let

$$\kappa_i(a) = C_i(\{a\})$$

if a is not worthless and let $\kappa_i(a) = 0$ if a is worthless.

It remains to show that κ_i is a feasible cutoff. For this, we use the following claim.

Claim 3. *Suppose t_i is a dichotomous type such that $f(t_i) \notin A(t_i)$. If f is 1-generation monotone, then $f(t_i) \in L^{\kappa_i}(A(t_i))$.*

Proof. Suppose t_i is a dichotomous type such that $f(t_i) = a \notin A(t_i)$. Assume, to the contrary, that $a \notin L^{\kappa_i}(A(t_i))$. This means $\kappa_i(a) > 0$ and, hence, a is not a worthless alternative. Consider a dichotomous type $\bar{t}_i$ such that $A(\bar{t}_i) = \{a\}$ and $v(\bar{t}_i) < \kappa_i(a)$. By definition, $f(\bar{t}_i) \neq a$. Hence, $t_i \in G_1^f(\bar{t}_i)$. By 1-generation monotonicity (GSS), t_i must satisfy itself. This is a contradiction. $\lhd$

Now, to see that κ_i is a feasible cutoff, assume, to the contrary, that it is not. Then for every alternative $a \in \mathcal{A}$, $\kappa_i(a) > 0$. Pick any $a \in \mathcal{A}$. Since $\kappa_i(a) > 0$, for any dichotomous type t_i such that $v(t_i) < \kappa_i(a)$ and $A(t_i) = \{a\}$, $f(t_i) \neq a$ (by Claim 1). By Claim 3, $f(t_i) \in L^{\kappa_i}(\{a\})$. But by our assumption, $L^{\kappa_i}(\{a\}) = \varnothing$. This is a contradiction.

We now prove another claim.

Claim 4. *Suppose t_i is a dichotomous type such that $f(t_i) \in A(t_i)$. If f is 2-generation monotone, then $\kappa_i(f(t_i)) \leq \kappa_i(a)$ for all $a \in A(t_i)$.*

Proof. Let t_i be a dichotomous type such that $A(t_i) = S$ and $f(t_i) = b \in S$. Choose $a \in S \setminus \{b\}$. Assume, to the contrary, that $\kappa_i(b) - \kappa_i(a) > \epsilon > 0$ for some ϵ. Consider two dichotomous types $\bar{t}_i$ and $\hat{t}_i$ such that

$$v(\bar{t}_i) = \kappa_i(b) - \tfrac{1}{2}\epsilon, \qquad A(\bar{t}_i) = \{b\}$$
$$v(\hat{t}_i) = \kappa_i(a) + \tfrac{1}{2}\epsilon, \qquad A(\hat{t}_i) = \{a\}.$$

By definition, $f(\bar{t}_i) \neq b$ and $f(\hat{t}_i) = a$. Hence, $t_i \in G_1^f(\bar{t}_i)$ and $\hat{t}_i \in G_2^f(\bar{t}_i)$. By 2-generation monotonicity, $\kappa_i(a) + \epsilon/2 \geq \kappa_i(b) - \epsilon/2$. Hence, $\kappa_i(b) - \kappa_i(a) \leq \epsilon$, which is a contradiction. $\triangleleft$

Claim 4 establishes that if for any dichotomous type t_i, we have $f(t_i) \in A(t_i)$, then $\kappa_i(f(t_i)) = \min_{a \in A(t_i)} \kappa_i(a)$. Hence, $f(t_i) \in W^{\kappa_i}(A(t_i))$. This establishes that if f is implementable, then it is cutoff-based.

We now show that if f is cutoff-based, then it is implementable. Let the feasible cutoff corresponding to f be κ_i. We show that f is 2-generation monotone. Consider t_i such that $f(t_i) \notin A(t_i)$ and let $s_i \in G_1^f(t_i)$. Assume, to the contrary, that $f(s_i) \notin A(s_i)$. By the definition of the cutoff-based rule, $f(s_i) \in L^{\kappa_i}(A(s_i))$. This implies that $\kappa_i(f(s_i)) = 0$. But $f(s_i) \in A(t_i)$ implies that $\min_{a \in A(t_i)} \kappa_i(a) = C^{\kappa_i}(A(t_i)) = 0$. Since, f is cutoff-based, this means that $f(t_i) \in A(t_i)$. This is a contradiction. So $f(s_i) \in A(s_i)$. Now, using the definition of the cutoff-based rule and the definition of $C^{\kappa_i}(\cdot)$, gives

$$v(s_i) \geq C^{\kappa_i}(A(s_i)) = \kappa_i(f(s_i)) \geq C^{\kappa_i}(A(t_i)) \geq v(t_i).$$

This shows that f is 1-generation monotone.

Now, consider $\bar{s}_i$ such that $\bar{s}_i \in G_2^f(t_i)$. Assume, to the contrary, that $f(\bar{s}_i) \notin A(\bar{s}_i)$. In that case, $\kappa_i(f(\bar{s}_i)) = 0$. But $f(\bar{s}_i) \in A(s_i)$ implies that $C^{\kappa_i}(A(s_i)) = 0$. But we know that s_i satisfies itself and, hence, $\kappa_i(f(s_i)) = C^{\kappa_i}(A(s_i)) = 0$ (by the definition of the cutoff-based rule). Then consider any type $\bar{t}_i$ with $A(\bar{t}_i) = \{f(s_i)\}$ and $0 < v(\bar{t}_i) < v(t_i)$. By definition, $f(\bar{t}_i) = f(s_i) \in A(t_i)$. Hence, $\bar{t}_i \in G_1^f(t_i)$. By 1-generation monotonicity, $v(\bar{t}_i) \geq v(t_i)$. This is a contradiction. Hence, $f(\bar{s}_i) \in A(\bar{s}_i)$. By the definition of the cutoff-based rule and the definition of $C^{\kappa_i}(\cdot)$, we have

$$v(\bar{s}_i) \geq \kappa_i(f(\bar{s}_i)) \geq C^{\kappa_i}(A(s_i)) = \kappa_i(f(s_i)) \geq C^{\kappa_i}(A(t_i)) \geq v(t_i).$$

This shows that f is 2-generation monotone. $\square$

4.3 *Discussions*

Why 3-cycle monotonicity? The 3-cycle monotonicity characterization critically relies on the richness assumption of dichotomous domains. Without the richness assumption, this need not hold.[7] Richness allows us to define payments as in (2). Without richness, the associated payments of an implementable allocation rule in a dichotomous domain can be quite complicated.

One natural question then is, *Why is 2-cycle monotonicity not sufficient in a rich dichotomous domain?* The proof of Theorem 3 sheds some light onto this. The proof shows what 1-generation monotonicity alone gives us and the additional implication of 2-generation monotonicity. One-generation monotonicity shows that the allocation rule must have cutoffs for each alternative (Claim 1) and that the cutoff for any acceptable set is the minimum over the cutoffs of alternatives in that acceptable set (Claim 2). Further, 1-generation monotonicity gives us that when a type is not satisfied (i.e., when the value is below the cutoff of its acceptable set), the alternative chosen by the allocation rule must be an alternative with zero cutoff (Claim 3).

However, when a type is satisfied, 1-generation monotonicity alone does not impose any restriction on what alternative in the acceptable set may be chosen by the allocation rule. Strong 1-generation monotonicity (equivalently, 2-cycle monotonicity) has some bite on which alternative must be chosen in this case. But it does not completely make the rule cutoff-based (necessary and sufficient for implementability). This is clearly illustrated in the example in Figure 1, where the allocation rule is strong 1-generation monotone, but still not implementable: this allocation rule chooses a when the acceptable set is $\{a, b\}$ and the value is above 5 but a is not the "cheapest" alternative.

Strong 1-generation monotonicity brings in NR on top of 1-generation monotonicity. In the example in Figure 1, NR imposes only some restrictions when the acceptable set contains alternative c. But for types that do not contain c, strong 1-generation monotonicity does not fix the outcome when such a type is satisfied. This makes the allocation rule not implementable. Alternatively, 2-generation monotonicity helps to fix this problem. When a dichotomous type is satisfied, it must be satisfied only by an alternative in the acceptable set whose cutoff value is the same as the cutoff value of the acceptable set (Claim 4).

Fixing the allocation rule in Figure 1. Using Theorem 3, we can now fix the allocation rule in Figure 1. Since this allocation rule already satisfies strong 1-generation monotonicity, the only modification we need to do is to assign the correct outcome using Theorem 3 when a dichotomous type is satisfied—when $\{a, b\}$ is the acceptable set. This is shown in Figure 2.

Generation number. A plausible conjecture is that in rich dichotomous domains, every implementable allocation rule has a generation number less than or equal to 2.

[7] In the working paper version of this paper (Mishra and Roy 2012), we discuss some dichotomous domains, including the single-minded domain, that are not rich. The 3-cycle monotonicity characterization does not hold in those domains. Further, we also identify a domain where the acceptable set of the agent always consists of one alternative. This domain is *essentially* one dimensional, and we show that 2-cycle monotonicity is sufficient in this domain.

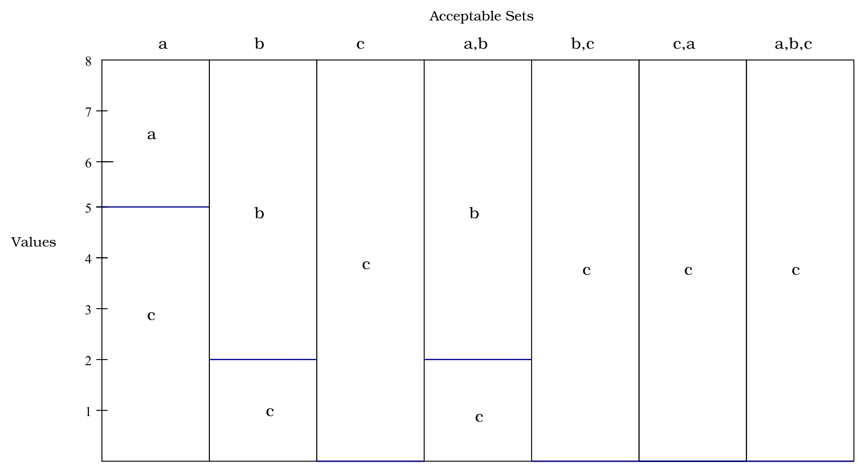

FIGURE 2. A cutoff-based allocation rule.

This conjecture is false. Consider a problem with four alternatives: $\mathcal{A} = \{a, b, c, d\}$. Let f be an allocation rule defined by the cutoff mapping κ_i: $\kappa_i(a) = 5$, $\kappa_i(b) = 3$, $\kappa_i(c) = 2$, $\kappa_i(d) = 0$. The allocation rule f is cutoff-based using the cutoffs defined in κ_i and, hence, it is implementable due to Theorem 3. However, consider the four types

$$s_i^0 = (2 - \epsilon, \{c\})$$
$$s_i^1 = (2 + \epsilon, \{b, c\})$$
$$s_i^2 = (3 + \epsilon, \{a, b\})$$
$$s_i^3 = (5 + \epsilon, \{a\}),$$

where $\epsilon \in (0, 2)$. By definition of the cutoff κ_i, we see that $f(s_i^0) = d$, $f(s_i^1) = c$, $f(s_i^2) = b$, and $f(s_i^3) = a$. Hence, $s_i^1 \in G_1^f(s_i^0)$, $s_i^2 \in G_2^f(s_i^0)$, and $s_i^3 \in G_3^f(s_i^0)$. So $\gamma^f \geq 3$.

4.4 *Revenue equivalence*

In this section, we establish that revenue equivalence holds in rich dichotomous domains. The seminal revenue equivalence result of Myerson (1981) has been extended to the multidimensional setup by many authors; see, for example, Milgrom and Segal (2002), Krishna and Maenner (2001), Chung and Olszewski (2007), and Heydenreich et al. (2009). These papers establish that every implementable allocation rule satisfies revenue equivalence if the domain satisfies certain assumptions. The assumptions in these papers require that the domain be connected.

However, our domain is not connected. To see this, consider an example with three alternatives: $\mathcal{A} = \{a, b, c\}$. Suppose all possible acceptable sets are permissible, i.e., $\Sigma = \{S : S \subseteq \mathcal{A}, S \neq \varnothing\}$. Suppose the value at any dichotomous type lies in $(0, \infty)$. Then the type space in $\mathbb{R}^3$ is as shown in Figure 3. It consists of seven open rays originating from the origin (but not including the origin). The positive parts of the three axes constitute three rays and they refer to those dichotomous types where there is a single acceptable alternative. The positive parts of the 45° degree rays in the xy, yz, and zx planes

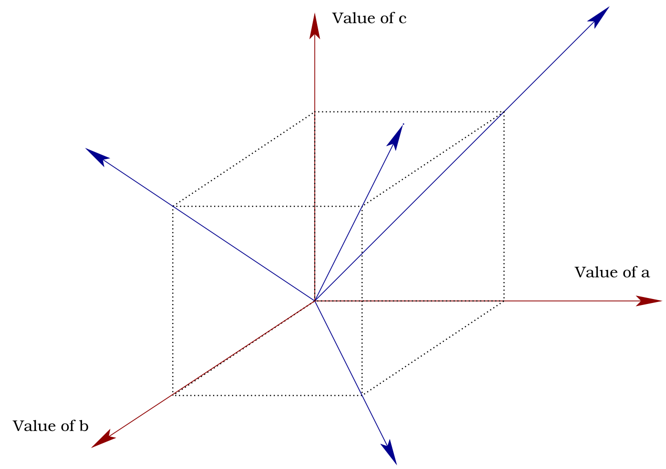

FIGURE 3. A rich dichotomous domain with three alternatives.

are three more rays and they refer to those dichotomous types where the acceptable set consists of any two alternatives. Finally, the positive part of the ray from the origin and passing through $(1, 1, 1)$ consists of all dichotomous types where the acceptable set is $\{a, b, c\}$. Note that this type space is not connected since the origin is not part of it.

Heydenreich et al. (2009) give a condition for the allocation rule (instead of domain) such that it satisfies revenue equivalence. We use their result to prove revenue equivalence in rich dichotomous domains.

THEOREM 4. *If $f : \mathcal{D}_i \to \mathcal{A}$ is an implementable allocation rule, where $\mathcal{D}_i$ is a rich dichotomous domain, and if p_i implements f, then*

$$p_i(t_i) = \kappa_i^f(f(t_i))\delta(f(t_i), t_i) + c_i \quad \forall t_i \in \mathcal{D}_i,$$

where c_i is a constant and κ_i^f is as defined in (2).

The proof of Theorem 4 is given in the Appendix. We note that Theorem 4 holds even without the (b) part of the richness assumption (Definition 6). The (a) part of the richness assumption is required since it makes the closure of the domain connected, which allows revenue equivalence to go through using a result in Heydenreich et al. (2009).

We use this revenue equivalence result in Section 6 to determine a revenue maximizing mechanism in a one-sided matching model with agents that have dichotomous types.

5. EXTENSION TO n AGENTS

In this section, we show how our results can be extended to a setting with more than one agent. Suppose $N = \{1, \ldots, n\}$ is the set of n agents. An allocation rule f in the dichotomous domain is now a mapping $f : \mathcal{D} \to \mathcal{A}^n$, where $\mathcal{D} = \mathcal{D}_1 \times \mathcal{D}_2 \times \cdots \times \mathcal{D}_n$ denotes the set of all dichotomous type profiles. Note that the outcome of an allocation rule is in $\mathcal{A}^n$. So an allocation rule specifies an alternative for each agent at every type profile. We denote the allocation of agent i at type profile t as $f_i(t) \in \mathcal{A}$. We assume *absence of allocative externality*, so the value of an agent is completely determined by his own allocation.

There may be feasibility constraints that link the allocations of different agents at each type profile. For instance, in the collective choice problems, such as the problems of hiring a staff jointly by departments and choosing a network to build, all agents must get the same alternative as allocation, i.e., for every type profile t, we must have $f_i(t) = f_j(t)$ for all $i, j \in N$. The richness restriction (b) in Section 4 (Definition 6) applies to the set of alternatives $\mathcal{A}$ and not to $\mathcal{A}^n$.

Alternatively, in private good problems, such as a single-minded combinatorial auction or matching with transfers, each agent i is faced with a set of alternatives $\mathcal{A}$. In the case of a single-minded combinatorial auction, $\mathcal{A}$ is the set of all subsets of objects. In the case of matching with transfers in the job market, the set of alternatives for a firm is the set of all job candidates. An allocation rule chooses an alternative in $\mathcal{A}$ for every agent such that it constitutes a feasible outcome, e.g., in case of matching, it is a feasible matching (no candidate is assigned more than one job). The richness restriction Definition 6(b) in applies to the set of alternatives $\mathcal{A}$ and not to $\mathcal{A}^n$. With this interpretation, all our definitions and results extend easily: we just need to add "for all t_{-i}" in all the definitions.

6. Application: Revenue maximizing matching with dichotomous preferences

In this section, we apply our results on characterizing implementable allocation rules in rich dichotomous domains. We derive an optimal mechanism in a one-sided matching problem where agents have dichotomous types. We assume that the set of alternatives is $\mathcal{A}$ and that this includes a worthless alternative a_0. We denote the set of alternatives without a_0 as $\mathcal{A}_0 \equiv \mathcal{A} \setminus \{a_0\}$. The interpretation of $\mathcal{A}_0$ can be a set of objects (time periods where an airline ticket is available or schools to which a student can be assigned, etc.). The worthless alternative a_0 can be interpreted as the alternative where an agent is not assigned any object. Let $N = \{1, \dots, n\}$ be the set of n agents. The acceptable set of each agent $i \in N$ is a subset of $\mathcal{A}_0$. Using our earlier notation, we let $\Sigma := \{A \subseteq \mathcal{A}_0 : A \neq \varnothing\}$. We assume that at any dichotomous type, the value of any agent $i \in N$ lies in the interval $V_i = (0, \beta_i)$, where $\beta_i \in \mathbb{R}_{++} \cup \{\infty\}$. This ensures that the type space of every agent is a rich dichotomous domain. We refer to this problem as *one-sided matching with dichotomous preferences*.

An allocation rule f is a mapping $f : \mathcal{D} \to \mathcal{A}^n$. So f assigns each agent an alternative in $\mathcal{A}$: this is a private good allocation problem. There may be feasibility constraints. For instance, there may be a finite number of units of every object. In the example of students matching to schools, a school may have a capacity constraint on the number of students they can take. In the example of agents assigned to different time periods of an airline, the number of tickets available in a time period may be finite. We denote such constraints on the outcome of f as $\mathcal{F}$ and we assume that there is no restriction on the number of agents who can be assigned the alternative a_0. An outcome of an allocation rule is an element of $\mathcal{A}^n$ that satisfies the feasibility constraints of $\mathcal{F}$; it is called an *assignment*.

We assume that the type of each agent's type is drawn independently as follows. The probability that $A \subseteq \mathcal{A}_0$ is the acceptable set of agent i is given by $h_i(A)$. The value of agent i is drawn using a distribution g_i with cumulative distribution function G_i. Note that we assume that the value of agent i is independent of his acceptable set. We assume that the hazard rate $g_i(v_i)/(1 - G_i(v_i))$ is nondecreasing in v_i. Let $w_i : V_i \to \mathbb{R}$ be the *virtual valuation function* of agent i, defined as

$$w_i(v_i) = v_i - \frac{1 - G_i(v_i)}{g_i(v_i)} \quad \forall v_i \in V_i.$$

Since the hazard rate is nondecreasing, the virtual valuation function is increasing.

Now fix an allocation rule f. We denote the alternative assigned to agent i at any type profile t as $f_i(t)$. Suppose f is implementable and $p \equiv (p_1, \dots, p_n)$ implements f. In such a case, we say that the mechanism (f, p) is dominant strategy incentive compatible (DSIC). Then the expected revenue in mechanism (f, p) is given by

$$\Pi(f, p) = \sum_{i \in N} E_t[p_i(t)],$$

where $E_t[\cdot]$ denotes the expectation over all the type profiles. A mechanism (f, p) is *individually rational* if at every type profile $t \in \mathcal{D}$, we have $v(t_i)\delta(f_i(t), t_i) - p_i(t) \geq 0$ for all $i \in N$.

Definition 9. A mechanism (f, p) is an *optimal mechanism* if it is DSIC, individually rational, and there does not exist another mechanism (f', p') such that (f', p') is DSIC, individually rational, and $\Pi(f', p') > \Pi(f, p)$.

Consider a DSIC mechanism (f, p) and a rich dichotomous type profile $t \equiv (t_1, \dots, t_n)$. By Theorem 4, the payment of agent $i \in N$ at type profile t is given by

$$p_i(t) = c_i(t_{-i}) + \kappa^f_{i,t_{-i}}(f_i(t))\delta(f_i(t), t_i), \tag{4}$$

where $\kappa^f_{i,t_{-i}}$ is the cutoff of agent i that corresponds to the allocation rule f (as defined in (2)) and $c_i : \mathcal{D}_{-i} \to \mathbb{R}$ is an arbitrary function. Using the definition of cutoff $\kappa^f_{i,t_{-i}}(f_i(t))$ and our characterization result of Theorem 3, we know that for any dichotomous type with $A(t_i)$ as an acceptable set, agent i is satisfied at all values above $\kappa^f_{i,t_{-i}}(f_i(t))$ and is not satisfied at all values below $\kappa^f_{i,t_{-i}}(f_i(t))$. Hence, we can write the payment of agent i at type profile t as

$$p_i(t) = c_i(t_{-i}) + v(t_i)\delta(f_i(t), t_i) - \int_0^{v(t_i)} \delta(f_i((x_i, A(t_i)), t_{-i}), (x_i, A(t_i)))\, dx_i, \tag{5}$$

where we write $(x_i, A(t_i))$ to denote a dichotomous type with value x_i and acceptable set $A(t_i)$. To see how (4) and (5) are equivalent, note that by our characterization in

Theorem 3 of implementable rule using cutoffs, we can conclude that the value of the integral in (5) is 0 if $\delta(f_i(t), t_i) = 0$ and is $[v(t_i) - \kappa^f_{i,t_{-i}}(f_i(t))]$ if $\delta(f_i(t), t_i) = 1$.

Once we have the expression for the payment in this form, we employ the methodology of Myerson (1981) to express the expected revenue in terms of virtual valuations. The expected payment of agent i in the DSIC mechanism (f, p) is given by

$$\begin{aligned}
\pi_i(f, p) &= E_{t_{-i}}\biggl[c_i(t_{-i}) + \sum_{A\in\mathcal{A}_0}\biggl[\int_0^{\beta_i} z_i\delta(f_i((z_i, A), t_{-i}), (z_i, A))g_i(z_i)\,dz_i \\
&\qquad - \int_0^{\beta_i}\biggl(\int_0^{z_i}\delta(f_i((x_i, A), t_{-i}), (x_i, A))\,dx_i\biggr)g_i(z_i)\,dz_i\biggr]h_i(A)\biggr] \\
&= E_{t_{-i}}\biggl[c_i(t_{-i}) + \sum_{A\in\mathcal{A}_0}\biggl[\int_0^{\beta_i} z_i\delta(f_i((z_i, A), t_{-i}), (z_i, A))g_i(z_i)\,dz_i \\
&\qquad - \int_0^{z_i}(1 - G_i(z_i))\delta(f_i((z_i, A), t_{-i}), (z_i, A))\,dz_i\biggr]h_i(A)\biggr] \\
&\qquad \text{(changing order of integration)} \\
&= E_{t_{-i}}\biggl[c_i(t_{-i}) \\
&\qquad + \sum_{A\in\mathcal{A}_0}\biggl[\int_0^{\beta_i}\biggl(z_i - \frac{1 - G_i(z_i)}{g_i(z_i)}\biggr)\delta(f_i((z_i, A), t_{-i}), (z_i, A))g_i(z_i)\,dz_i\biggr]h_i(A)\biggr].
\end{aligned}$$

Hence, the expected revenue in the DSIC mechanism (f, p) is given by

$$\Pi(f, p) = \sum_{i\in N} E_{t_{-i}}\biggl[c_i(t_{-i}) + \sum_{A\in\mathcal{A}_0}\biggl[\int_0^{\beta_i} w_i(z_i)\delta(f_i((z_i, A), t_{-i}), (z_i, A))g_i(z_i)\,dz_i\biggr]h_i(A)\biggr].$$

Note that if (f, p) is individually rational, then for every $i \in N$ and every t_{-i}, we have $c_i(t_{-i}) \le 0$. If (f, p) is individually rational and we want to maximize the expected revenue, then we must have $c_i(t_{-i}) = 0$ for all $i \in N$ and for all t_{-i}. Using this, the expression of the expected revenue in the DISC mechanism (f, p) is reduced to

$$\begin{aligned}
\Pi(f, p) &= \sum_{i\in N} E_{t_{-i}}\biggl[\sum_{A\in\mathcal{A}_0}\biggl[\int_0^{\beta_i} w_i(z_i)\delta(f_i((z_i, A), t_{-i}), (z_i, A))g_i(z_i)\,dz_i\biggr]h_i(A)\biggr] \\
&= E_t\biggl[\sum_{i\in N} w_i(v(t_i))\delta(f_i(t), t_i)\biggr].
\end{aligned}$$

If we sidestep the fact that f needs to be 2-generation monotone (for it to be implementable), the above expression can be maximized by doing pointwise maximization. So at every type profile, we look at those agents whose virtual values are nonnegative. For any agent whose virtual valuation is not positive, he is assigned the alternative a_0. Else, an alternative $a^i \in \mathcal{A}$ is assigned to agent i such that $(a^1, \ldots, a^n) \in \mathcal{F}$ (i.e., a feasible allocation), and the sum of virtual values of all the agents who have positive virtual values is maximized from this allocation. Formally, at every type profile $t \in \mathcal{D}$, let

$W(t) := \{i \in N : w_i(v(t_i)) > 0\}$. The optimal allocation rule f^* is defined as follows. For every type profile t, denote by $\mathcal{A}^n(t) \subseteq \mathcal{A}^n$ the set of feasible assignments where each agent $i \notin W(t)$ is assigned the worthless alternative a_0. In other words, at every type profile t, if we take any $a \in \mathcal{A}^n(t)$, then for every $i \notin W(t)$, we have $a^i = a_0$, where a^i is the alternative assigned to agent i in assignment a. Then f^* is defined as any allocation rule that satisfies

$$f^*(t) \in \underset{(a^1,\dots,a^n)\in\mathcal{A}^n(t)}{\arg\max} \Bigg[\sum_{i\in W(t)} w_i(v(t_i))\delta(a^i, A(t_i))\Bigg], \tag{6}$$

where we assume $f_i^*(t) = a_0$ if $\delta(f_i^*(t), t_i) = 0$, i.e., if an agent is unsatisfied, then he is assigned a_0 (note that this does not influence the outcome of the maximization). We show that f^* is implementable.

Proposition 3. *The allocation rule f^* is implementable.*

Proof. Using Theorem 2, we only need to show that f^* satisfies 2-generation monotonicity. Fix an agent i and type profile t_{-i} of other agents. Let t_i be a type of agent i such that $f_i^*(t_i, t_{-i}) = a_0$. Suppose s_i is a first generation type of t_i at t_{-i}. Then $f_i^*(s_i, t_{-i}) \in A(t_i)$. This implies that $f_i^*(s_i, t_{-i}) \neq a_0$. Hence, we have $f_i^*(s_i, t_{-i}) \in A(s_i)$; if $f_i^*(s_i, t_{-i}) \notin A(s_i)$, then by definition of f^*, $f_i^*(s_i, t_{-i}) = a_0$, which is not possible. This establishes GSS. Note that by definition, $w_i(v(s_i)) > 0$. If $w_i(v(t_i)) \leq 0$, then $w_i(v(s_i)) > 0$ implies that $v(s_i) > v(t_i)$ (since the virtual valuation function is increasing). This establishes MON when $w_i(v(t_i)) \leq 0$. Now, assume $w_i(v(t_i)) > 0$. Let $W(t_{-i}) := \{j \in N : w_j(v(t_j) > 0\}$. Denote the allocation of any agent $j \in N$ at type profile (t_i, t_{-i}) as T_j and that at type profile (s_i, t_{-i}) as S_j. Using the definition of f^*, we can write the two inequalities

$$w_i(v(t_i))\delta(T_i, t_i) + \sum_{j\in W(t_{-i})} w_j(v(t_j))\delta(T_j, t_j) \geq w_i(v(t_i))\delta(S_i, t_i) + \sum_{j\in W(t_{-i})} w_j(v(t_j))\delta(S_j, t_j)$$

$$w_i(v(s_i))\delta(S_i, s_i) + \sum_{j\in W(t_{-i})} w_j(v(t_j))\delta(S_j, t_j) \geq w_i(v(s_i))\delta(T_i, s_i) + \sum_{j\in W(t_{-i})} w_j(v(t_j))\delta(T_j, t_j).$$

Adding these two inequalities, and using the fact that $\delta(T_i, t_i) = 0$, $\delta(S_i, t_i) = 1$, and $\delta(S_i, s_i) = 1$, we get

$$w_i(v(s_i)) - w_i(v(s_i))\delta(T_i, s_i) \geq w_i(v(t_i)).$$

Since $w_i(v(t_i)) > 0$, the above inequality is feasible only if $\delta(T_i, s_i) = 0$ and $v(s_i) \geq v(t_i)$. This establishes MON. Hence, f^* is 1-generation monotone.

Now, for 2-generation monotonicity, consider s_i' which, is a second generation type of t_i at t_{-i}. Suppose s_i' satisfies s_i, where s_i is a first generation type of t_i. Note that since

$f_i^*(s_i', t_{-i}) \in A(s_i)$, $f_i^*(s_i', t_{-i}) \neq a_0$, and by the definition of f^*, we have $f_i^*(s_i', t_{-i}) \in A(s_i')$. This establishes GSS.

Since $f_i^*(s_i', t_{-i}) \in A(s_i')$, this means $w_i(v(s_i')) > 0$. If $w_i(v(t_i)) \leq 0$, then $v(s_i') > v(t_i)$ (since the virtual valuation function is increasing). Consider the case where $w_i(v(t_i)) > 0$. Suppose the allocation of any agent $j \in N$ in type profile (t_i, t_{-i}) is T_j, in type profile (s_i, t_{-i}) is S_j, and in type profile (s_i', t_{-i}) is S_j'. Using the definition of f^*, we get the inequalities

$$w_i(v(t_i))\delta(T_i, t_i) + \sum_{j \in W(t_{-i})} w_j(v(t_j))\delta(T_j, t_j) \geq w_i(v(t_i))\delta(S_i, t_i) + \sum_{j \in W(t_{-i})} w_j(v(t_j))\delta(S_j, t_j)$$

$$w_i(v(s_i))\delta(S_i, s_i) + \sum_{j \in W(t_{-i})} w_j(v(t_j))\delta(S_j, t_j) \geq w_i(v(s_i))\delta(S_i', s_i) + \sum_{j \in W(t_{-i})} w_j(v(t_j))\delta(S_j', t_j)$$

$$w_i(v(s_i'))\delta(S_i', s_i') + \sum_{j \in W(t_{-i})} w_j(v(t_j))\delta(S_j', t_j) \geq w_i(v(s_i'))\delta(T_i, s_i') + \sum_{j \in W(t_{-i})} w_j(v(t_j))\delta(T_j, t_j).$$

Using the facts that $\delta(S_i, s_i) = \delta(S_i, t_i) = \delta(S_i', s_i) = \delta(S_i', s_i') = 1$ and $\delta(T_i, t_i) = 0$, and adding the above inequalities we get that

$$w_i(v(s_i')) - w_i(v(s_i'))\delta(T_i, s_i') \geq w_i(v(t_i)).$$

Since $w_i(v(t_i)) > 0$, the above inequality is feasible only if $\delta(T_i, s_i') = 0$ and $v(s_i') \geq v(t_i)$. This establishes MON. Hence, f^* is 2-generation monotone. □

This shows that f^* along with the cutoff payment defined in Proposition 2 is the optimal mechanism. This is summarized in the following theorem.

Theorem 5. *In the one-sided matching problem with dichotomous preferences, the optimal mechanism is given by* (f^*, p^*)*, where* f^* *is defined as in (6), and for every* $t \in \mathcal{D}$ *and every* $i \in N$*,* $p_i^*(t) = \kappa_{i,t_{-i}}^{f^*}(f_i^*(t))\delta(f_i^*(t), t_i)$*, where* $\kappa_{i,t_{-i}}^{f^*}$ *is defined as in (2).*

Remark. Notice that the optimal mechanism is independent of the probability distribution of acceptable sets. Intuitively, the payments are determined by cutoffs of *values*. Revenue maximization is, therefore, related to how values are distributed. Since we assume that the value distribution is independent of the distribution of acceptable sets, the optimal mechanism is dependent only on the distribution of values.

Remark. A special case of the optimal mechanism occurs when there is just one agent. This problem is referred to as the revenue maximization of a multiple good monopolist seller and is recognized as a hard problem if the type of the buyer is multidimensional (Manelli and Vincent 2007). Theorem 5 says that if there is one agent i, then the optimal mechanism is to set a reserve price equal to r^*, which solves $r^* = (1 - G_i(r^*))g_i(r^*)$; there is a unique solution to this if the hazard rate is nondecreasing. Agent i is satisfied by allocating *any* alternative in his acceptable set if his value is above r^*; he is not satisfied by allocating a_0 if his value is less than or equal to r^*.

Remark. Unlike Myerson (1981), who searched for an optimal mechanism in the single object auction case over all Bayesian incentive compatible and randomized mechanisms, we are searching over all DSIC and deterministic mechanisms. Most of the literature on optimal mechanism design in multidimensional type spaces also considers Bayes–Nash randomized implementation (for example, Iyengar and Kumar 2008, and Pai and Vohra 2010). For single object auctions, this restriction is without loss of generality since the optimal mechanism is a DSIC and deterministic mechanism; see a more general result for the single object auction case in Manelli and Vincent (2010). However, we do not know if we enlarge our search to include Bayesian incentive compatible and randomized mechanisms, whether we will improve expected revenues in this setting.

7. Conclusion

The seminal paper of Myerson (1981) contains three important results in mechanism design in quasilinear environments for the single object auction case: (1) a characterization of implementable allocation rules; (2) an illustration of revenue equivalence; (3) a derivation of optimal mechanism. Each of these results has been generalized to various multidimensional settings. We contribute to this literature by extending these results to specific dichotomous domains.

Our general methodology in this paper is to derive a simplification of cycle monotonicity in specific multidimensional dichotomous domains. Whether we can derive similar simplifications in other interesting nonconvex domains and then use them to derive an easy characterization of implementability remains an open question. It will also be interesting to extend our results with randomization and/or consideration of relaxed forms of implementability like Bayes–Nash implementability.

Appendix: Omitted proofs

Proof of Lemma 1. Let $K = |\mathcal{A}|$, and let f be a K-cycle monotone allocation rule. Consider any cycle $C \equiv (t_i^1, \dots, t_i^k, t_i^1)$ in the type graph. The proof is by induction on k. If $k \le K$, then by definition, this cycle has nonnegative length. Suppose $k > K$ and assume that all cycles with less than k nodes have nonnegative length. Since $k > K$, there are two types t_i^h and t_i^j in the cycle C such that $f(t_i^h) = f(t_i^j)$. Note that the lengths of the edges (t_i^h, t_i^j) and (t_i^j, t_i^h) are both zero. Assume without loss of generality $h < j$. We consider two cases.

Case 1. If $h = j - 1$, then note that the length of the edge (t_i^h, t_i^{j+1}) is the same as the length of the edge (t_i^j, t_i^{j+1}). Hence, the length of the cycle $C' \equiv (t_i^1, \dots, t_i^h, t_i^{j+1}, \dots, t_i^k, t_i^1)$ is the same as the length of the cycle C. But C' has one less node than C. By our induction hypothesis, the length of cycle C' is nonnegative. So the length of cycle C is nonnegative.

Case 2. If $h = 1$ and $j = k$, then we repeat Case 1, but this time we consider the cycle $C' \equiv (t_i^2, \dots, t_i^k, t_i^2)$.

Case 3. In this case, there is at least one node between t_i^h and t_i^k, and at least one node between t_i^k and t_i^h in cycle C. We can now break the cycle C into two parts, $C^1 \equiv (t_i^1, \dots, t_i^h, t_i^j, t_i^{j+1}, \dots, t_i^k, t_i^1)$ and $C^2 \equiv (t_i^h, t_i^{h+1}, \dots, t_i^j, t_i^h)$. Since $f(t_i^j) = f(f_i^h)$, the edges (t_i^h, t_i^j) and (t_i^j, t_i^h) have zero length. Hence, the total length of both cycles C^1 and C^2 combined is equal to the length of cycle C. Further, C^1 and C^2 have less than k number of nodes. By our induction hypothesis, both C^1 and C^2 have nonnegative length. Hence, the length of the cycle C is nonnegative. $\square$

Proof of Proposition 1. Fix a positive integer $K \geq 2$ and an allocation rule f. Suppose f is K-cycle monotone. To show that f is strong $(K-1)$-generation monotone, consider any type t_i such that t_i is not satisfied (if no such t_i exists, then we are done vacuously). Pick any $t_i^k \in G_k^f(t_i)$, where $k \leq (K-1)$. We show that f satisfies GSS, MON, and NR by using induction on k.

For $k = 1$, consider the 2-cycle (t_i, t_i^1, t_i). The length of the edge from t_i^1 to t_i is $-v(t_i)$. Hence, the length of the edge from t_i to t_i^1 is at least $v(t_i)$. But the length of the edge from t_i to t_i^1 is

$$v(t_i^1)\big[\delta(f(t_i^1), t_i^1) - \delta(f(t_i), t_i^1)\big].$$

This length is at least $v(t_i)$ only if $\delta(f(t_i^1), t_i^1) = 1$ (GSS), $\delta(f(t_i), t_i^1) = 0$ (NR), and $v(t_i^1) \geq v(t_i)$ (MON).

Now, assume that f satisfies GSS, MON, and NR for all $k < r \leq (K-1)$. We show that for any $t_i^r \in G_r^f(t_i)$, we have that t_i^r is satisfied, $v(t_i^r) \geq v(t_i)$, and t_i does not satisfy t_i^r. We pick $t_i^1, t_i^2, \dots, t_i^{r-1}$ such that $t_i^j \in G_j^f(t_i)$ for all $j \in \{1, \dots, r-1\}$ and $\delta(f(t_i^j), t_i^{j-1}) = 1$ for all $j \in \{1, \dots, r-1\}$, where $t_i^0 = t_i$. By our induction hypothesis, $\delta(f(t_i^j), t_i^j) = 1$ (GSS) for all $j \in \{1, \dots, r-1\}$. As a result, for any $j \in \{2, \dots, r\}$, the length of the edge (t_i^j, t_i^{j-1}) is zero. So, the length of the cycle $C \equiv (t_i, t_i^r, t_i^{r-1}, \dots, t_i^1, t_i)$ is

$$v(t_i^r)\big[\delta(f(t_i^r), t_i^r) - \delta(f(t_i), t_i^r)\big] + v(t_i)\big[\delta(f(t_i), t_i) - \delta(f(t_i^1), t_i)\big].$$

By our assumption, $\delta(f(t_i^1), t_i) = 1$ and $\delta(f(t_i), t_i) = 0$. Hence, the length of the cycle C is

$$v(t_i^r)\big[\delta(f(t_i^r), t_i^r) - \delta(f(t_i), t_i^r)\big] - v(t_i).$$

By our assumption, the length of the cycle C is nonnegative. This can be made nonnegative only if $v(t_i^r) \geq v(t_i)$ (MON), $\delta(f(t_i^r), t_i^r) = 1$ (GSS), and $\delta(f(t_i), t_i^r) = 0$ (NR). This concludes the proof that f is strong $(K-1)$-generation monotone.

Now, for the converse, suppose f is strong $(K-1)$-generation monotone. We show that f is K-cycle monotone. We do the proof in several steps.

Step 1. We show that f is 2-cycle monotone. Consider a cycle (s_i, t_i, s_i), and assume, to the contrary, that it has negative length. Then at least one of the edges in the cycle has negative length. Without loss of generality, let the length of the edge from s_i to t_i be negative. Then $v(t_i)[\delta(f(t_i), t_i) - \delta(f(s_i), t_i)] = -v(t_i) < 0$. This implies that $\delta(f(t_i), t_i) = 0$ but $\delta(f(s_i), t_i) = 1$. Hence, t_i is not satisfied, but $s_i \in G_1^f(t_i)$. By strong generation monotonicity, s_i is satisfied, $v(s_i) \geq v(t_i)$, and s_i is not satisfied by t_i. This implies that the length of the edge (t_i, s_i) is $v(s_i) \geq v(t_i)$. Hence, the length of the 2-cycle is nonnegative, which is a contradiction.

Step 2. We consider any cycle $C \equiv (t_i^1, \dots, t_i^K, t_i^1)$ such that t_i^j is satisfied for all $j \in \{1, \dots, K\}$. In that case, the length of any arbitrary edge (t_i^j, t_i^{j+1}) of this cycle is $v(t_i^{j+1})[\delta(f(t_i^{j+1}), t_i^{j+1}) - \delta(f(t_i^j), t_i^{j+1})] \geq 0$, where we denote $(j+1) \equiv 1$ if $j = K$. Hence, the cycle C has nonnegative length.

Step 3. We consider any cycle with K nodes where exactly one node, say t_i, is not satisfied and all other nodes are satisfied. Denote this cycle by $C \equiv (t_i, t_i^{K-1}, t_i^{K-2}, \dots, t_i^1, t_i)$. Note that the length of any edge (t_i^j, t_i^{j-1}) for any $j \in \{2, \dots, K\}$, where $t_i^K \equiv t_i$, is equal to

$$v(t_i^{j-1})\big[\delta(f(t_i^{j-1}), t_i^{j-1}) - \delta(f(t_i^j), t_i^{j-1})\big] = v(t_i^{j-1})\big[1 - \delta(f(t_i^j), t_i^{j-1})\big],$$

which is equal to $v(t_i^{j-1})$ if t_i^{j-1} is not satisfied by t_i^j and is equal to zero if t_i^{j-1} is satisfied by t_i^j. Thus, all such edges have nonnegative length.

Now, consider the edge (t_i^1, t_i). The length of this edge is

$$v(t_i)\big[\delta(f(t_i), t_i) - \delta(f(t_i^1), t_i)\big] = -v(t_i)\delta(f(t_i^1), t_i).$$

If $\delta(f(t_i^1), t_i) = 0$, then the length of the cycle C is nonnegative. Else, the length of the edge (t_i^1, t_i) is $-v(t_i)$, and it is the only negative length edge of C. In this case, $t_i^1 \in G_1^f(t_i)$. By MON, $v(t_i^1) \geq v(t_i)$. Now we evaluate the length of edge (t_i^2, t_i^1). If $\delta(f(t_i^2), t_i^1) = 0$, then the length of the edge (t_i^2, t_i^1) is $v(t_i^1) \geq v(t_i)$ and, hence, the length of the cycle C is nonnegative. Else, $\delta(f(t_i^2), t_i^1) = 1$ implies that $t_i^2 \in G_2^f(t_i)$. By generation monotonicity, $v(t_i^2) \geq v(t_i)$. Continuing in this manner, we either find a node/type t_i^j, where $j \in \{2, \dots, K-1\}$, such that $v(t_i^j) \geq v(t_i)$ and $\delta(f(t_i^j), t_i^{j-1}) = 0$ or we reach an edge (t_i, t_i^{K-1}) with $t_i^{K-1} \in G_{K-1}^f(t_i)$. The length of this edge is

$$v(t_i^{K-1})\big[\delta(f(t_i^{K-1}), t_i^{K-1}) - \delta(f(t_i), t_i^{K-1})\big].$$

By strong $(K-1)$-generation monotonicity, $v(t_i^{K-1}) \geq v(t_i)$, $\delta(f(t_i^{K-1}), t_i^{K-1}) = 1$ and $\delta(f(t_i), t_i^{K-1}) = 0$. Hence, the length of this edge is $v(t_i^{K-1}) \geq v(t_i)$. This shows that the length of the cycle C is nonnegative.

Step 4. In this step, we show that for any s_i and t_i such that s_i and t_i are not satisfied, the length of the edge (s_i, t_i) is zero. Consider any 2-cycle $C \equiv (s_i, t_i, s_i)$ such that s_i and t_i are not satisfied. The length of C is zero. To see this, note that the length of C is nonnegative since f is 2-cycle monotone by our induction hypothesis. Further, the length of C is

$$v(t_i)\big[\delta(f(t_i), t_i) - \delta(f(s_i), t_i)\big] + v(s_i)\big[\delta(f(s_i), s_i) - \delta(f(t_i), s_i)\big]$$
$$= -\big[v(t_i)\delta(f(s_i), t_i) + v(s_i)\delta(f(t_i), s_i)\big] \leq 0.$$

This shows that the length of C is zero. Hence, the length of the edges from s_i to t_i and from t_i to s_i are both zero. This shows that in any cycle where all the nodes are not satisfied, the length of the edges in this cycle must be zero.

Step 5. Now we show that the length of a particular cycle is nonnegative. A cycle $(t_i^1, \dots, t_i^h, t_i^1)$ is an *interior cycle* if $h \geq 3$, t_i^1 and t_i^h are not satisfied, and t_i^j is satisfied for all $j \in \{2, \dots, h-1\}$. Consider an interior cycle $C \equiv (t_i^1, \dots, t_i^K, t_i^1)$ with $K > 2$. We show that its length is nonnegative. Since C is an interior cycle, assume without loss of generality that t_i^1 and t_i^K are not satisfied, but t_i^j is satisfied for all $j \in \{2, \dots, K-1\}$. Since f is 2-cycle monotone (by our induction hypothesis), the length of edge (t_i^K, t_i^1) is zero; this follows from Step 4. The length of the edge (t_i^{K-1}, t_i^K) is

$$v(t_i^K)\big[\delta(f(t_i^K), t_i^K) - \delta(f(t_i^{K-1}), t_i^K)\big].$$

Since $\delta(f(t_i^K), t_i^K) = 0$, the length of the edge (t_i^{K-1}, t_i^K) is nonpositive. Now, consider any edge (t_i^j, t_i^{j+1}) in cycle C such that $(t_i^j, t_i^{j+1}) \notin \{(t_i^{K-1}, t_i^K), (t_i^K, t_i^1)\}$, where $(j+1) \equiv 1$ if $j = K$. By definition, $\delta(f(t_i^{j+1}), t_i^{j+1}) = 1$. Hence, the length of this edge is nonnegative.

Hence, the only edge in C that may have a negative length is (t_i^{K-1}, t_i^K). Suppose the length of edge (t_i^{K-1}, t_i^K) is negative. In that case, $\delta(f(t_i^{K-1}), t_i^K) = 1$, and the length of the edge is $-v(t_i^K)$. We show that some other edge in C has a length greater than or equal to $v(t_i^K)$.

Note that $t_i^{K-1} \in G_1^f(t_i^K)$. By strong $(K-1)$-generation monotonicity, $v(t_i^{K-1}) \geq v(t_i^K)$ and $\delta(f(t_i^{K-1}), t_i^{K-1}) = 1$. The length of the edge (t_i^{K-2}, t_i^{K-1}) is either zero or $v(t_i^{K-1})$. If it is $v(t_i^{K-1})$, we are done. Else, $\delta(f(t_i^{K-2}), t_i^{K-1}) = 1$ implies that $t_i^{K-2} \in G_2^f(t_i^K)$. By strong $(K-1)$-generation monotonicity again, $v(t_i^{K-2}) \geq v(t_i^K)$. Continuing in this manner, we either find an edge whose length is greater than or equal to $v(t_i^K)$ or reach the edge (t_i^1, t_i^2) with zero edge length, and $t_i^1 \in G_{K-1}^f(t_i^K)$. In that case, strong $(K-1)$-generation monotonicity implies that $\delta(f(t_i^1), t_i^1) = 1$, which is a contradiction.

Step 6. In the final step, we consider a cycle C with K nodes, where $K > 2$. If C has less than or equal to one unsatisfied type, then we are done by Steps 2 and 3. Else, we break this cycle into subcycles where each cycle starts from an unsatisfied type and ends with another unsatisfied type. So C is broken into cycles $C_1 = (t_i^1, \dots, t_i^{j_1}, t_i^1)$, $C_2 = (t_i^{j_1}, \dots, t_i^{j_2}, t_i^{j_1}), \dots,$ $C_\ell = (t_i^{j_{\ell-1}}, \dots, t_i^1, t_i^{j_{\ell-1}})$, and $C_{\ell+1} = (t_i^1, t_i^{j_1}, t_i^{j_2}, \dots, t_i^{j_{\ell-1}}, t_i^1)$, where the length of $C_{\ell+1}$ is zero since all its types are unsatisfied (Step 4). Note that the sum of the lengths of these cycles gives the length of the cycle C. Since each of the cycles C_p, where $p \in \{1, \dots, \ell\}$, is an interior cycle, they have nonnegative length by Step 5. Hence, the length of C is nonnegative. □

Proof of Theorem 4. We use a result in Heydenreich et al. (2009) to prove our theorem. We say that an implementable allocation rule f satisfies revenue equivalence if for any two payment rules p_i and p'_i, there exists a constant c_i such that for all s_i, $p_i(s_i) = p'_i(s_i) + c_i$. Heydenreich et al. (2009) characterize allocation rules that satisfy revenue equivalence. To describe and use their result, we need the following notation.

Consider the type graph that corresponds to rich dichotomous domain $\mathcal{D}_i$ and allocation rule f. For any pair of types s_i and t_i, a path in the type graph from s_i to t_i is a sequence of distinct nodes $P \equiv (s_i, s_i^1, \dots, s_i^k, t_i)$. Denote by $l(P)$ the length of a path P. Let P_{s_i,t_i} be the set of all paths from s_i to t_i. Define $\text{dist}^f(s_i, t_i) := \inf_{P \in P_{s_i,t_i}} l(P)$, i.e., the length of the shortest path from s_i to t_i in the type graph. If f is implementable, then it can be shown that for all $s_i, t_i \in \mathcal{D}_i$, $\text{dist}^f(s_i, t_i)$ is a real number and $\text{dist}^f(s_i, t_i) + \text{dist}^f(t_i, s_i) \geq 0$ (Heydenreich et al. 2009).

Heydenreich et al. (2009) show that an allocation rule f satisfies revenue equivalence if and only if for all $s_i, t_i \in \mathcal{D}_i$, we have $\text{dist}^f(s_i, t_i) + \text{dist}^f(t_i, s_i) = 0$. We show that this condition holds in our domain. Consider two dichotomous types $s_i, t_i \in \mathcal{D}_i$. Assume, to the contrary, $\text{dist}^f(s_i, t_i) + \text{dist}^f(t_i, s_i) = \epsilon > 0$. Consider two types $\bar{s}_i$ and $\bar{t}_i$ such that $A(\bar{s}_i) = A(s_i)$, $A(\bar{t}_i) = A(t_i)$, $v(\bar{s}_i) < v(s_i)$, $v(\bar{t}_i) < v(t_i)$, and $v(\bar{s}_i) + v(\bar{t}_i) < \epsilon$. Note that because of this,

$$\ell(\bar{s}_i, \bar{t}_i) + \ell(\bar{t}_i, \bar{s}_i) < \epsilon.$$

We now look at the shortest paths between s_i and $\bar{s}_i$. Note that s_i and $\bar{s}_i$ have the same acceptable set. If $\delta(f(s_i), s_i) = \delta(f(\bar{s}_i), \bar{s}_i)$, then $\ell(s_i, \bar{s}_i) + \ell(\bar{s}_i, s_i) = 0$ and, hence, $\text{dist}^f(s_i, \bar{s}_i) + \text{dist}^f(\bar{s}_i, s_i) = 0$. Else, since $v(\bar{s}_i) < v(s_i)$, by Theorem 3, $\delta(f(s_i), s_i) = 1$ and $\delta(f(\bar{s}_i), \bar{s}_i) = 0$. Let the cutoff corresponding to $A(s_i)$ be $C(A(s_i))$. Then we have $v(s_i) \geq C(A(s_i)) \geq v(\bar{s}_i)$. Consider the type s'_i such that $A(s'_i) = A(s_i)$ and $v(s'_i) = C(A(s_i))$. We consider two cases.

Case 1. Suppose $f(s'_i) \in A(s_i)$. Then $v(\bar{s}_i) < C(A(s_i))$. Choose another type s''_i such that $A(s''_i) = A(s_i)$ and $v(s''_i) = v(s'_i) - \epsilon' > v(\bar{s}_i)$. The 2-cycle between s_i and s'_i has length zero since $A(s_i) = A(s'_i)$ and $\delta(f(s_i), s_i) = \delta(f(s'_i), s'_i) = 1$. Further, the 2-cycle between s''_i and $\bar{s}_i$ has length zero since $A(\bar{s}_i) = A(s''_i)$ and $\delta(f(s''_i), s''_i) = \delta(f(\bar{s}_i), \bar{s}_i) = 0$. Finally, the 2-cycle between s'_i and s''_i has length equal to

$$v(s'_i)\big[\delta(f(s'_i), s'_i) - \delta(f(s''_i), s'_i)\big] + v(s''_i)\big[\delta(f(s''_i), s''_i) - \delta(f(s'_i), s''_i)\big] = v(s'_i) - v(s''_i) = \epsilon'.$$

Hence, the sum of lengths of the path $(s_i, s'_i, s''_i, \bar{s}_i)$ and the path $(\bar{s}_i, s''_i, s'_i, s_i)$ is ϵ'.

Case 2. Suppose $f(s'_i) \notin A(s_i)$. Then $v(s_i) > C(A(s_i))$, and we can choose s''_i to be a type such that $A(s''_i) = A(s_i)$ and $v(s''_i) = v(s'_i) + \epsilon' < v(s_i)$. Using a similar argument to Case 1, we can show that the length of the 2-cycle between s'_i and s''_i is ϵ'. Further, the 2-cycles between s_i and s''_i and between s'_i and $\bar{s}_i$ are zero. Hence, the sum of lengths of the path $(s_i, s''_i, s'_i, \bar{s}_i)$ and the path $(\bar{s}_i, s'_i, s''_i, s_i)$ is ϵ'.

So, we conclude that $\text{dist}^f(s_i, \bar{s}_i) + \text{dist}^f(\bar{s}_i, s_i) \leq \epsilon'$. Since ϵ' can be chosen arbitrarily close to zero, we conclude that $\text{dist}^f(s_i, \bar{s}_i) + \text{dist}^f(\bar{s}_i, s_i) = 0$. Because of this, there is a path P from s_i to $\bar{s}_i$ and another path P' from s_i' to s_i such that $l(P) + l(P')$ is arbitrarily close to zero.

Using a similar argument, we can show that $\text{dist}^f(t_i, \bar{t}_i) + \text{dist}^f(\bar{t}_i, t_i) = 0$. Because of this, there is a path Q from t_i to $\bar{t}_i$ and another path Q' from $\bar{t}_i$ to t_i such that $l(Q) + l(Q')$ is arbitrarily close to zero.

Hence, the total length of $l(P) + \ell(\bar{s}_i, \bar{t}_i) + l(Q') + l(Q) + \ell(\bar{t}_i, \bar{s}_i) + l(P')$ is less than ϵ. This contradicts the fact that $\text{dist}^f(s_i, t_i) + \text{dist}^f(t_i, s_i) = \epsilon$. The proof now concludes by observing from Proposition 2 that one payment rule p^* that implements f is $p_i^*(t_i) = \kappa_i^f(f(t_i))\delta(f(t_i), t_i)$ for all t_i. □

References

Aggarwal, Gagan and Jason D. Hartline (2006), "Knapsack auctions." In *Proceedings of the 17th Annual ACM-SIAM Symposium on Discrete Algorithms*, 1083–1092, Association for Computing Machinery, New York. [433]

Archer, Aaron and Robert Kleinberg (2008), "Truthful germs are contagious: A local to global characterization of truthfulness." In *Proceedings of the 9th ACM Conference on Electronic Commerce*, 21–30, Association for Computing Machinery, New York. [434]

Archer, Aaron, Christos Papadimitriou, Kunal Talwar, and Éva Tardos (2003), "An approximate truthful mechanism for combinatorial auctions with single parameter agents." In *Proceedings of the 14th Annual ACM-SIAM Symposium on Discrete Algorithms*, 205–214, Society for Industrial and Applied Mathematics, Philadelphia. [433]

Archer, Aaron and Éva Tardos (2001), "Truthful mechanisms for one-parameter agents." In *Proceedings of the 42nd IEEE Symposium on Foundations of Computer Science*, 482–491, IEEE Computer Society, Washington, DC. [433, 435]

Armstrong, Mark (1996), "Multiproduct nonlinear pricing." *Econometrica*, 64, 51–75. [435]

Ashlagi, Itai, Mark Braverman, Avinatan Hassidim, and Dov Monderer (2010), "Monotonicity and implementability." *Econometrica*, 78, 1749–1772. [432, 434, 437]

Babaioff, Moshe, Ron Lavi, and Elan Pavlov (2005), "Mechanism design for single-value domains." In *Proceedings of the 20th National Conference on Artificial Intelligence, Volume 1*, 241–247, AAAI Press, Cambridge, Massachusetts. [432, 434, 435]

Berger, André, Rudolf Müller, and Seyed Hossein Naeemi (2010), "Path-monotonicity and incentive compatibility." Research Memorandum RM/10/035, Maastricht University School of Business and Economics. [434]

Bikhchandani, Sushil, Shurojit Chatterji, Ron Lavi, Ahuva Mu'alem, Noam Nisan, and Arunava Sen (2006), "Weak monotonicity characterizes deterministic dominant-strategy implementation." *Econometrica*, 74, 1109–1132. [432, 434]

Blackorby, Charles and Dezsö Szalay (2007), "Multidimensional screening, affiliation, and full separation." Warwick Economic Research Papers 802, University of Warwick. [435]

Blumrosen, Liad and Noam Nisan (2007), "Combinatorial auctions." In *Algorithmic Game Theory* (Noam Nisan, Tim Roughgarden, Éva Tardos, and Vijay V. Vazirani, eds.), 267–300, Cambridge University Press, New York. [435]

Bogomolnaia, Anna and Hervé Moulin (2004), "Random matching under dichotomous preferences." *Econometrica*, 72, 257–279. [432]

Bogomolnaia, Anna, Hervé Moulin, and Richard Strong (2005), "Collective choice under dichotomous preferences." *Journal of Economic Theory*, 122, 165–184. [432]

Carbajal, Juan Carlos and Jeffrey C. Ely (2012), "Mechanism design without revenue equivalence." *Journal of Economic Theory*, 148, 104–133. [434]

Chung, Kim-Sau and Wojciech Olszewski (2007), "A non-differentiable approach to revenue equivalence." *Theoretical Economics*, 2, 469–487. [451]

Crawford, Vincent P. and Elsie Marie Knoer (1981), "Job matching with heterogeneous firms and workers." *Econometrica*, 49, 437–450. [433]

Cuff, Katherine, Sunghoon Hong, Jesse A. Schwartz, Quan Wen, and John A. Weymark (2012), "Dominant strategy implementation with a convex product space of valuations." *Social Choice and Welfare*, 39, 567–597. [434]

Dhangwatnotai, Peerapong, Shahar Dobzinski, Shadding Dughmi, and Tim Roughgarden (2008), "Truthful approximation schemes for single-parameter agents." In *Proceedings of the 49th IEEE Symposium on Foundations of Computer Science*, 15–24, IEEE Press, New York, New York. [433]

Goldberg, Aandrew V. and Jason D. Hartline (2005), "Collusion-resistant auctions for single-parameter agents." In *Proceedings of the 16th Annual ACM-SIAM Symposium on Discrete Algorithms*, 620–629, Society for Industrial and Applied Mathematics, Philadelphia. [433]

Gui, Hongwei, Rudolf Müller, and Rakesh V. Vohra (2004), "Characterizing dominant strategy mechanisms with multidimensional types." Unpublished paper. [433, 437]

Heydenreich, Birgit, Rudolf Müller, Marc Uetz, and Rakesh V. Vohra (2009), "Characterization of revenue equivalence." *Econometrica*, 77, 307–316. [433, 434, 451, 452, 462]

Iyengar, Garud and Anuj Kumar (2008), "Optimal procurement mechanisms for divisible goods with capacitated suppliers." *Review of Economic Design*, 12, 129–154. [435, 458]

Jehiel, Philippe, Benny Moldovanu, and Ennio Stacchetti (1999), "Multidimensional mechanism design for auctions with externalities." *Journal Economic Theory*, 85, 258–293. [434]

Kos, Nenad and Matthias Messner (2013), "Extremal incentive compatible transfers." *Journal of Economic Theory*, 148, 134–164. [434]

Krishna, Vijay and Eliot Maenner (2001), "Convex potentials with an application to mechanism design." *Econometrica*, 69, 1113–1119. [451]

Lehmann, Daniel, Liadan Ita O'Callaghan, and Yoav Shoham (2002), "Truth revelation in approximately efficient combinatorial auctions." *Journal of the ACM*, 49, 577–602. [433, 434]

Manelli, Alejandro M. and Daniel R. Vincent (2007), "Multidimensional mechanism design: Revenue maximization and the multiple-good monopoly." *Journal of Economic Theory*, 137, 153–185. [436, 458]

Manelli, Alejandro M. and Daniel R. Vincent (2010), "Bayesian and dominant strategy implementation in the independent private-values model." *Econometrica*, 78, 1905–1938. [458]

Milgrom, Paul R. and Ilya Segal (2002), "Envelope theorems for arbitrary choice sets." *Econometrica*, 70, 583–601. [451]

Mishra, Debasis and Souvik Roy (2012), "Implementation in multidimensional dichotomous domains." Unpublished paper, Indian Statistical Institute. [441, 450]

Müller, Rudolf, Andrés Perea, and Sascha Wolf (2007), "Weak monotonicity and Bayes-Nash incentive compatibility." *Games and Economics Behavior*, 61, 344–358. [434]

Myerson, Roger B. (1981), "Optimal auction design." *Mathematics of Operations Research*, 6, 58–73. [433, 435, 451, 455, 458]

Pai, Mallesh M. and Rakesh Vohra (2010), "Optimal auctions with financially constrained bidders." Unpublished paper. [436, 458]

Rahman, David (2011), "Detecting profitable deviations." Unpublished paper. [434]

Rochet, Jean-Charles (1987), "A necessary and sufficient condition for rationalizability in a quasi-linear context." *Journal of Mathematical Economics*, 16, 191–200. [432, 433, 437]

Rochet, Jean-Charles and Lars A. Stole (2003), "The economics of multidimensional screening." In *Advances in Economics and Econometrics, Volume I* (Mathias Dewatripont, Lars Peter Hansen, and Stephen J. Turnovsky, eds.), 150–197, Cambridge University Press, Cambridge. [435]

Rockafellar, R. Tyrrell (1970), *Convex Analysis*. Princeton University Press, Princeton, New Jersey. [433, 437]

Roth, Alvin E., Tayfun Sönmez, and M. Utku Unver (2005), "Pairwise kidney exchange." *Journal of Economic Theory*, 125, 151–188. [432]

Saks, Michael and Lan Yu (2005), "Weak monotonicity suffices for truthfulness on convex domains." In *Proceedings of 6th ACM Conference on Electronic Commerce*, 286–293, ACM Press, New York. [432, 434]

Vohra, Rakesh V. (2011), *Mechanism Design: A Linear Programming Approach*. Cambridge University Press, Cambridge. [434]

Vorsatz, Marc (2007), "Approval voting on dichotomous preferences." *Social Choice and Welfare*, 28, 127–141. [432]

Vorsatz, Marc (2008), "Scoring rules on dichotomous preferences." *Social Choice and Welfare*, 31, 151–162. [432]

Submitted 2012-5-14. Final version accepted 2012-6-25. Available online 2012-6-25.