Abstract

In the design of incentive compatible mechanisms, a common approach is to enforce incentive compatibility as constraints in programs that optimize over feasible mechanisms. Such constraints are often imposed on sparsified representations of the type spaces, such as their discretizations or samples, in order for the program to be manageable. In this work, we explore limitations of this approach, by studying whether all dominant strategy incentive compatible mechanisms on a set \( T \) of discrete types can be extended to the convex hull of \( T \).

Dobzinski, Fu and Kleinberg (2015) answered the question affirmatively for all settings where types are single dimensional. It is not difficult to show that the same holds when the set of feasible outcomes is downward closed. In this work we show that the question has a negative answer for certain non-downward-closed settings with multi-dimensional types. This result should call for caution in the use of the said approach to enforcing incentive compatibility beyond single-dimensional preferences and downward closed feasible outcomes.

Introduction

Mechanism design studies optimization problems with private inputs from strategic agents. An agent’s input, known as her type, is her private information including her valuation for the social outcomes, which are to be decided upon by the mechanism. A mechanism needs to solicit such information to achieve certain goals, e.g. maximizing welfare, revenue, surplus or fairness measures. It needs to provide its participants with correct incentives, via both social outcomes and payments, so that the agents find it in their best interests to reveal their types.

Dominant strategy incentive compatibility (DSIC) is one of the strongest and most widely used solution concepts that guarantee such incentives. Under DSIC, every participant, no matter what type she possesses and no matter what types the other participants report to the mechanism, will maximize her utility by revealing her true type. Not only is this a strong guarantee for the mechanism designer that true information should be reported and optimized over, it also alleviates the burden of strategizing from the participating agents — telling truth is a dominant strategy regardless of the other agents’ types or strategies. Partly thanks to this strong incentive guarantee, the two fundamental auctions, namely, the VCG auction that maximizes social welfare [Vickrey 1961; Clarke 1971; Groves 1973], and Myerson’s auction that maximizes expected revenue for selling a single item [Myerson 1981], have been foundational in both the theory and practice of mechanism design.

As the scope of mechanism design expands beyond the classical settings, incentive compatible mechanisms that are optimal for various objective often lack the simple structures that characterize the VCG and Myerson’s mechanisms. By and large, there have been two approaches to the design of incentive compatible mechanisms. The first approach focuses on classes of mechanisms that, by their simple structures, have obvious incentive guarantees. For example, in a multi-item auction, a sequential pricing mechanism puts prices on items and asks each agent in turn to choose her favorite items that remain; the bidders are not asked about their values, and choosing utility-maximizing items (according to their true values) is the obvious strategy to adopt (see, e.g., Chawla et al. 2010; Chawla, Malec, and Sivan 2010; Feldman, Gravin, and Lucier 2015). Another example is to optimize over parameterized “VCG-like” mechanisms which inherit incentive properties from the VCG mechanism (e.g. Sandholm and Likhodedov 2015). This approach is often used to search for mechanisms whose performance is a factor away from being optimal, since the optimal mechanism or its very close approximations are often not within the class of mechanisms being searched over.

The second approach forgoes structures that are easily interpretable, and exhaustively searches for the optimal mechanism. This is exemplified by solving mathematical programs (typically linear or convex programs) whose feasible regions contain the set of all incentive compatible mechanisms (see, e.g. Conitzer and Sandholm 2002; Dobzinski, Fu, and Kleinberg 2015; Daskalakis, Deckelbaum, and Tzamos 2017; Feng and Hartline 2018). Typically, incentive requirements are hardwired as constraints in such programs. Difficulty arises in the second approach when one would like to adopt strong incentive guarantees such as DSIC, which need at least one constraint per profile of types to specify. When the space of possible types is a continuum, this gives rise to uncountably many constraints. While this does not always make the program impossible to solve it
considerably complicates the task. One way to work around this is to discretize the type space and only impose incentive compatible (IC) constraints on the set of discrete types used to represent the type space. Discretization is also embodied in the idea of a given prior distribution over a set of discrete types, on which the optimization can then be based (e.g., Conitzer and Sandholm 2002; Dobzinski, Fu, and Kleinberg 2015). The most common motivation for such prior distributions is that they naturally result from samples (e.g. from past observations or market research) from an underlying distribution, whereas the true distribution may be supported on a continuum of types. This approach motivates the question we study in this work.

Questions we study. In this work we aim to answer the question: when one has a mechanism that is DSIC on a discretized subset of a type space, can one always find a mechanism that has the same behavior on the subset and yet is DSIC on the whole type space? To make the question more concrete, we study the natural case where the whole type space is the convex hull of the discrete subset. To make the presentation easier, in the following we denote by $T$ the discrete subset of types, and $\text{Conv}(T)$ its convex hull.

We consider the question a fundamental one for the second approach to mechanism design that we described above. When one optimizes for a mechanism with IC constraints imposed only on $T$, if the resulting mechanism cannot be extended to the original type space, it loses incentive guarantees when put to use.

One objection may be that, given a mechanism that is DSIC on $T$, one may always run it on $\text{Conv}(T)$, by restricting the “bidding language”, so that a type in $\text{Conv}(T)$ but not in $T$ has to report a type in $T$. Such mechanisms, however, may lose the incentive guarantee which makes DSIC mechanisms attractive in the first place. Unless one can show that agents with types not in $T$ have a dominant strategy in such mechanisms, such agents need to strategize over which types in $T$ to report, depending on the types and strategies of their opponents. In the scenario where $T$ is a set of samples from a continuous distribution, the vast majority of types may not be in $T$ and have no incentive guarantee, which is clearly undesirable. In settings where the agents’ types are single dimensional, Dobzinski, Fu, and Kleinberg (2015) showed that “restricting the bidding language” does turn any mechanism DSIC on $T$ into a mechanism DSIC on any superset of $T$: each type in the superset has a dominant strategy, and by the revelation principle this gives rise to a DSIC mechanism that extends the given mechanism’s behavior on $T$. To the best of our knowledge, no such guarantees are known beyond single dimensional settings.

Our Results. For agents with multi-dimensional types, we first give a condition under which any DSIC mechanism on $T$ can be extended to a DSIC mechanism on $\text{Conv}(T)$, via an argument that is different from Dobzinski, Fu, and Kleinberg’s yet still straightforward (Theorem 1). In particular, the condition is satisfied whenever the set of feasible outcomes is downward closed (Theorem 1).

Our main result, however, is a construction of a set $T$ of multi-dimensional types and a DSIC mechanism on it, for which we show that no DSIC mechanism on $\text{Conv}(T)$ can output the same social outcomes on types in $T$. The impossibility result stands even if the extension mechanism is allowed to be randomized. This shows that, without conditions such as single-dimensional types or downward closed set of feasible outcomes, designing incentive compatible mechanisms by focusing on a discrete subset of types can be a questionable approach to designing mechanisms for the whole type space — there may not be any mechanism DSIC on the whole type space which behave the same way on the subset.

Near the end, we give a multi-dimensional setting where the expected revenue of a mechanism with only correct incentives for a set $T$ of types can be unboundedly more than the revenue of a mechanism for $\text{Conv}(T)$. This example is much less involved than our main result, because revenue optimal mechanisms are meaningful only when they do not overcharge any reported type and guarantee non-negative utility. This constraint can be much more stringent when imposed for all types in $\text{Conv}(T)$ than for $T$ only.

Related Works

For multi-dimensional preferences, an allocation rule for any fixed agent is implementable if and only if it satisfies the so-called cyclic monotonicity property (Rochet 1987), when the type space is convex, it turns out the weaker condition of weak monotonicity suffices for implementability (Saks and Yu 2005; Archer and Kleinberg 2014). It is notable that the two solution concepts we compare in Section precisely correspond to the case where the type space is convex and that where it is not. However, nowhere in our arguments do we make use of this beautiful fact.

Another closely related body of work, is the literature on automated mechanism design. In automated mechanism design, mechanisms are optimized for the setting and objective automatically using information about agents’ type distributions. When this work was introduced by Conitzer and Sandholm (2002) the input for this problem was an explicit description of the agents distributions, however recent work has moved towards replacing this explicit description with samples from the agents type distribution. Likhodedov and Sandholm 2004, Likhodedov and Sandholm 2005 Sandholm and Likhodedov 2015). Our work highlights how interpolating between discrete samples can effect not only the objective but the implementability of the mechanism itself. Luckily, the research on sample based automated mechanism design is able to avoid the pitfalls of only having discrete samples. They do this either by optimizing over parameterized families of mechanisms which are guaranteed to be implementable on the entire typespace or by working in settings where the addition of new types has no effect on the objective (i.e. downward closed settings, see Theorem 1) Sandholm and Likhodedov 2015, Guo and Conitzer 2010, Balcan, Sandholm, and Vitercik 2018, Morgenstern and Roughgarden 2016). However, our work points out some difficulties that might arise if one wishes to take a more general, non-parameterized approach to au-
tomated mechanism design in settings which are not downward closed.

There are a variety of well-studied settings that are not downward closed in which extending a type space to its convex hull could cause problems. One commonly studied non-downward closed setting arises from the job scheduling problem introduced in [Nisan and Ronen (2001)] and later built upon by [Chawla et al. (2013)] and [Ashtiani, Dobzinski, and Lavi (2012)], since in this problem every job must eventually be scheduled and the set of feasible solutions is not downward closed. Another example of a non-downward closed setting is one-sided matching markets in which every agent must be matched with exactly one good. An example of a one-sided matching market is the fair housing allocation problem from Devanur, Mihail, and Vazirani (2005) also not downward closed.

**Preliminaries**

We consider a setting with \( N \) agents where each agent \( i \) has a private type \( t_i \) from her type space \( T_i \subseteq \mathbb{R}^n_i \). The type profile \( t = (t_1, \ldots, t_N) \) denotes the vector of all agents’ types, from the joint type space \( T := \prod T_i \).

We adopt the standard shorthand notation to write \( t_{-i} := (t_1, \ldots, t_{i-1}, t_{i+1}, \ldots, t_n) \) from \( T_{-i} := \prod_{j \neq i} T_j \).

An outcome (or, interchangeably, allocation) for agent \( i \) lies in \( R^N_i \); for an outcome \( x_i \), the agent with type \( t_i \) has value \( (t_i, x_i) = \sum_{j=1}^m t_{ij} x_{ij} \). A social outcome is denoted by a vector \( (x_1, \ldots, x_N) \in \mathbb{R}^N \). The set of all feasible social outcomes (or allocations) is denoted \( \mathcal{F} \subseteq \mathbb{R}^N \).

For example, in a single-item auction, \( m = 1 \), each \( t_i \in \mathbb{R}_+ \) represents agent \( i \)'s value for the item, and \( \mathcal{F} \subseteq \mathbb{R}_+ \) is the set of all integral points in \( \{ (x_1, \ldots, x_N) \in \mathbb{R}^N : \sum_{i=1}^N x_{ij} \leq 1, j = 1, 2, \ldots, m \} \).

**Mechanisms.** A (direct revelation) mechanism consists of an allocation rule \( x : T \rightarrow \mathcal{F} \) and a payment rule \( p : T \rightarrow \mathbb{R}^N \). The mechanism elicits type reports from the agents, and on reported type profile \( t \), decides on an allocation \( x(t) \in \mathcal{F} \), with each agent \( i \) making a payment of \( p_i(t) \). In general, allocation rules can be randomized, in which case \( x(t) \) is a randomized variable supported on \( \mathcal{F} \). \( x(\cdot) \) induces allocation rule \( x_i(\cdot) \) for each agent \( i \); for all \( t \), \( x_i(t) \in \mathbb{R}^N \) is the vector consisting of the \( (i-1)m + 1 \)-st to the \( im \)-th coordinates in \( x(t) \). When \( x(t) \) is a random variable, so are \( x_i(t) \)’s.

When \( x(t) \) is deterministic, we write \( x(t) = y \in \mathcal{F} \) as a shorthand for \( \Pr[x(t) = y] = 1 \).

Agents have quasi-linear utilities, that is, when reporting type \( t_i' \), agent \( i \)'s utility is \( E[(t_i, x_i(t_i', t_{-i}))] - p_i(t_i, t_{-i}) \) (where the expectation is taken over the randomness in \( x(t) \)).

A mechanism is dominant strategy incentive compatible (DSIC) if, for all \( t \in T \), and for all \( t_i' \in T_i \), \( E[(t_i, x_i(t_i, t_{-i}))] - p_i(t_i, t_{-i}) \geq E[(t_i, x_i(t_i', t_{-i}))] - p_i(t_i', t_{-i}) \). An allocation rule \( x \) is said to be DSIC implementable or simply DSIC if there is a payment rule \( p \) such that \( (x, p) \) is a DSIC mechanism. In this case, we say \( x \) is implemented by payment rule \( p \).

**Extensions.** Given a subset \( S \subseteq \mathbb{R}^n \), we denote by \( \text{Conv}(S) \) the convex hull of \( S \).

**Definition 1.** An allocation rule \( \tilde{x} : \text{Conv}(T) \rightarrow \mathcal{F} \) is an extension of an allocation rule \( x : T \rightarrow \mathcal{F} \) if for all \( t \in T \), \( \tilde{x}(t) \) has the same distribution as \( x(t) \). Similarly, a payment rule \( \tilde{p} : \text{Conv}(T) \rightarrow \mathbb{R}^N \) is an extension of payment rule \( p : T \rightarrow \mathbb{R}^N \) if for all \( t \in T \), \( \tilde{p}(t) = p(t) \).

In Definition 1 if \( x(\cdot) \) is deterministic, then \( \tilde{x}(\cdot) \) being an extension simply means \( \tilde{x}(t) = x(t) \) for all \( t \in T \).

**Downward closed settings.** The feasible allocation set \( \mathcal{F} \) is downward closed if \( y \in \mathcal{F} \) entails \( x \in \mathcal{F} \) for all \( x \preceq y \), where \( x \preceq y \) denotes \( x_j \leq y_j \) for \( j = 1, \ldots, mN \).

**Weak monotonicity.** A well-known necessary condition for an allocation rule to be DSIC implementable is weak monotonicity:

**Definition 2.** An allocation rule \( x : T \rightarrow \mathcal{F} \) is weakly monotone if for each agent \( i \), any \( t_i, t_i' \in T_i \) and \( t_{-i} \in T_{-i} \),

\[
E[(t_i - t_i', x_i(t_i, t_{-i}) - x_i(t_i', t_{-i}))] \geq 0.
\]

**Proposition 1** (see, e.g., Saks and Yu (2005)). An allocation rule is implementable only if it is weakly monotone.

In convex, weak monotonicity is also a sufficient condition for DSIC implementability.

**Revenue.** A mechanism \((x, p)\) is ex post individually rational (IR) if for each agent \( i \) and for every \( t \in T \), \( (t_i, x_i(t)) - p_i(t) \geq 0 \).

Given a distribution \( D \) on \( T \) and a mechanism \((x, p)\) that is DSIC and ex post IR, the expected revenue of the mechanism is \( \mathbb{E}_{t \sim D} \sum p_i(t) \). The optimal revenue is the maximum expected revenue achievable among all DSIC, ex post IR mechanisms.

**DSIC Convex Extensions**

Before presenting our main result on the impossibility to extend DSIC allocation rules, we first complement [Dobzinski, Fu, and Kleinberg (2015)]’s result in single-dimensional setting with a simple observation in multi-dimensional preference settings: whenever the feasible allocation space is downward closed, any DSIC allocation rule on a type space can be extended to its convex hull by another DSIC allocation rule.

**Theorem 1.** If the set of feasible allocations \( \mathcal{F} \) is downward closed, for any DSIC allocation rule \( x \) on a type space \( T \), there is a DSIC extension \( \tilde{x} \) of \( x \) on \( \text{Conv}(T) \). If \( x \) is implemented with a payment rule \( p \), \( \tilde{x} \) can be implemented by an extension \( \tilde{p} \) of \( p \). If \( p \) is individually rational on \( T \), so is \( \tilde{p} \) on \( \text{Conv}(T) \).
If we do not require the statement about individual rationality, extensibility is guaranteed by an even weaker condition, which we call single swap feasible.

**Definition 3** (Single swap feasible). A feasible allocation set $\mathcal{F}$ is single swap feasible (SSF) if for every agent $i$ there exists an allocation $x^{ssf}(i) \in \mathcal{F}$ such that for any $x^i \in \mathcal{F}$, $(x^i, x^{ssf}(i)) \in \mathcal{F}$.

Intuitively, $x^{ssf}(i)$ is a feasible allocation vector such that if we replace the $i$th element of this vector with the $j$th element from any other feasible allocation the resulting allocation is still feasible. If $\mathcal{F}$ is a product space or is downward closed, it must be SSF.

**Theorem 2.** If the set of feasible allocations $\mathcal{F}$ is SSF, any DSIC allocation rule $x$ on a type space $T$, there is a DSIC extension $\tilde{x}$ of $x$ on $\text{Conv}(T)$.

The proofs for both Theorem 1 and Theorem 2 can be found in the supplementary materials. The main result of this paper is that without this condition a DSIC extension may not exist.

**Theorem 3.** There is a two agent type space $T^{\text{rand}}$ with a DSIC allocation rule $x^{\text{rand}}$, such that $x^{\text{rand}}$ cannot be extended by a DSIC allocation rule to $\text{Conv}(T^{\text{rand}})$.

We prove the theorem in two steps. We first present a set with three-dimensional preferences for which we show the lack of a payment identity`a la Myerson (Myerson 1981). In order to argue that any extension of an allocation rule is not DSIC, one has to either check the many cyclic (or weak) monotonicity conditions, or show that no payment rule can support the extension in a DSIC mechanism. We designed $T^{\text{det}}$ and $x^{\text{det}}$ carefully so that the allocations “lock” the payment rules.

**Lemma 1.** For any allocation rule $\tilde{x}$ that is an extension of $x^{\text{det}}$, if $\tilde{x}$ can be implemented by a DSIC mechanism with payment rule $p$, then for any $V \in \{A, B, C, D\}$, $p_1(A, V) = p_1(B, V) = p_1(C, V) = p_1(D, V)$, and $p_2(V, A) = p_2(V, B) = p_2(V, C) = p_2(V, D)$.

**Proof.** We prove the lemma for agent 1, and the statement for agent 2 follows by symmetry.

By DSIC, for any $V \in T^{\text{det}}$, we have

$$\langle A, x_1^{\text{det}}(A, V) \rangle - p_1(A, V) \geq \langle A, x_1^{\text{det}}(C, V) \rangle - p_1(C, V);$$

$$\langle B, x_1^{\text{det}}(B, V) \rangle - p_1(B, V) \geq \langle B, x_1^{\text{det}}(A, V) \rangle - p_1(A, V);$$

$$\langle C, x_1^{\text{det}}(C, V) \rangle - p_1(C, V) \geq \langle C, x_1^{\text{det}}(B, V) \rangle - p_1(B, V).$$

Note that

$$\langle A, x_1^{\text{det}}(A, V) \rangle = \langle A, x_1^{\text{det}}(C, V) \rangle;$$

$$\langle C, x_1^{\text{det}}(C, V) \rangle = \langle C, x_1^{\text{det}}(B, V) \rangle;$$

$$\langle B, x_1^{\text{det}}(B, V) \rangle = \langle B, x_1^{\text{det}}(A, V) \rangle.$$
Lemma 2. If $\tilde{x}$ is a deterministic DSIC extension of $x^{\text{det}}$, then for $t_1 = \frac{1}{3}A + \frac{1}{3}B + \frac{1}{3}D$, and any $V_2 \in T^{\text{det}}$, $\tilde{x}_1(t_1, V_2) \in \{\tilde{x}_1(A, V_2), \tilde{x}_1(B, V_2), \tilde{x}_1(C, V_2), \tilde{x}_1(D, V_2)\}$.

Proof. For the sake of contradiction, assume $\tilde{x}$ is a DSIC extension of $x^{\text{det}}$, implementable by payment rule $p$, and for $t_1$ and $V_2 \in T^{\text{det}}$, $\tilde{x}_1(t_1, V_2) = x^{\text{det}}_1(V_2')$ for some $(V_1', V_2') \in T^{\text{det}}$ and $V_2' \neq V_2$.

For any $V_2$, one of $x^{\text{det}}_1(A, V_2)$ and $x^{\text{det}}_1(B, V_2)$ gives an equal positive value to both $A$ and $B$. Let $V^*_1$ be the type that induces this equally valued allocation. (For example, if $V_2 = B$, then $x^{\text{det}}_1(A, B) = [2, 0, 2]$ and $x^{\text{det}}_1(B, B) = [2, 2, 0]$. Both $A$ and $B$ have the same value for $x^{\text{det}}_1(B, B)$ and so type $B$ would be $V^*_1$.)

Observe that, for the allocation $x^{\text{det}}_1(V_1, V_2')$, $V_1$ has positive value ($V_1, x^{\text{det}}_1(V_1, V_2')$, and no other type has higher value for it. Therefore, type $t_1$ has value at most $(\frac{2}{3}V_1 + \frac{1}{3}D, x^{\text{det}}_1(V_1, V_2'))$ for the allocation. In order for $t_1$ to have no incentive to misreport $V^*_1$, we must have

$$\frac{2}{3}V_1 + \frac{1}{3}D, x^{\text{det}}_1(V_1, V_2') - p_1(t_1, V_2) \geq \frac{2}{3}V^*_1 + \frac{1}{3}D, x^{\text{det}}_1(V^*_1, V_2) - p_1(V^*_1, V_2).$$

On the other hand, in order for type $V_1$ not to have incentive for deviating to $t_1$, we have

$$\langle V_1, x^{\text{det}}_1(V_1, V_2) \rangle = \langle V^*_1, x^{\text{det}}_1(V^*_1, V_2) \rangle - p_1(V^*_1, V_2);$$

where for the equality we used the fact that the value obtained by reporting truthfully is the same for every type in $\{A, B, C\}$ given a fixed type of the opponent, and that $p_1(V_1, V_2) = p_1(V^*_1, V_2)$ by Lemma 1.

Similarly, in order for type $D$ not to have incentive for deviating to $t_1$, we have

$$\langle D, x^{\text{det}}_1(D, V_2) \rangle - p_1(D, V_2) = \langle D, x^{\text{det}}_1(V^*_1, V_2) \rangle - p_1(D, V_2);$$

where the equality for the case $t_1$ was the same for all allocations given a fixed type of the opponent, and that $p_1(D, V_2) = p_1(V_1, V_2)$ by Lemma 1. Crucially, 2 and 3 cannot both be tight, because by construction, for any $V_2 \neq V_2$, $\langle V_1, x^{\text{det}}_1(V_1, V_2) - x^{\text{det}}_1(V_1, V_2) \rangle = \frac{2}{3}(D, x^{\text{det}}_1(D, V_2) - x^{\text{det}}_1(V_1, V_2)) \neq 0$.

Therefore, $\frac{2}{3} \cdot 2 + \frac{1}{3} \cdot 3$ gives

$$\frac{2}{3}V^*_1 + \frac{1}{3}D, x^{\text{det}}_1(V^*_1, V_2) - p_1(V^*_1, V_2) \geq \frac{2}{3}V_1 + \frac{1}{3}D, x^{\text{det}}_1(V_1, V_2) - p_1(t_1, V_2),$$

which contradicts 1.

By the same reasoning as for Lemma 1 the following lemma follows from Lemma 2.

Lemma 3. If $\tilde{x}$ is a deterministic DSIC extension of $x^{\text{det}}$, implementable by payment rule $p$, then for any $t_1$ in the interior of $\text{Conv}(T^{\text{det}})$ and any $V_2 \in T^{\text{det}}$, $p_1(t_1, V_2) = p_1(A, V_2)$.

We are now ready to prove Theorem 4.

Proof of Theorem 4. Suppose $\tilde{x}$ is a deterministic DSIC extension of $x^{\text{det}}$. We show a contradiction by showing that $\tilde{x}$ must violate weak monotonicity.

Consider $t_1 = \frac{1}{3}A + \frac{1}{3}B + \frac{1}{3}D$ and when agent 2’s type is $A$. Since $t_1$ could report any type in $T^{\text{det}}$, she has as options $x^{\text{det}}(A, A), x^{\text{det}}(B, A), x^{\text{det}}(C, A)$ and $x^{\text{det}}(D, A)$, all at the same price by Lemma 3. By Lemma 2 these are also all the allocations she could possibly get.

Since $[1, 1, 0]$ is the only allocation for which both types $A$ and $B$ have positive value, it is $t_1$’s preferred allocation, i.e., $\tilde{x}_1(t_1, A)$ must be $[1, 1, 0]$. This in turn implies $\tilde{x}_2(t_1, A) = [1, 1, 0]$. (Recall that $\tilde{x}$ is a product space, and the only allocation in which agent 1 gets $[1, 1, 0]$ is $x^{\text{det}}(A, A) = [1, 1, 0, 1, 1, 0, 1, 1, 0]$.)

Similarly, one can show $\tilde{x}_1(t_1, D) = [4, 4, 0]$, which implies $\tilde{x}_2(t_1, D) = [2, 2, 0]$ or $[4, 4, 0]$. But in either case, weak monotonicity is violated for agent 2’s types $A$ and $D$. For example, if $\tilde{x}_2(t_1, D) = [2, 2, 0]$, we have $\langle A, [1, 1, 0] \rangle + \langle D, [2, 2, 0] \rangle < \langle A, [2, 2, 0] \rangle + \langle D, [1, 1, 0] \rangle$.

Therefore, no deterministic DSIC extension of $x^{\text{det}}$ is possible.

Non-existence of randomized extensions

We need a more convoluted construction and a more careful argument to prove the impossibility of extensions that are possibly randomized. We build on $T^{\text{rand}}$ and $x^{\text{det}}$ to construct $T^{\text{rand}}$ and $x^{\text{rand}}$ and prove Theorem 3.

We first raise types in $T^{\text{det}}$ to a space of seven dimensions. Define $A' = [1, 0, 0, 0, 0, 0, 0], B' = [1, 1, 0, 0, 0, 0, 0], C' = [0, 0, 1, 0, 0, 0, 0]$ and $D' = \frac{1}{2}(A' + B' + C')$. For ease of notation, we define a mapping $\text{det} : \{A', B', C', D'\} \rightarrow T^{\text{det}}$, with $\text{det}(A') = A, \text{det}(B') = B, \text{det}(C') = C$ and $\text{det}(D') = D$. We also introduce four new types, $E' = [0, 0, 1, 0, 0, 0, 0], F' = [0, 0, 0, 0, 1, 0, 0], G' = [0, 0, 0, 0, 0, 0, 0]$, and $H' = [0, 0, 0, 0, 0, 0, 1]$. Define $T^{\text{rand}} = \{A', B', C', D', E', F', G', H'\}$, and $T^{\text{rand}} = T^{\text{rand}} \times T^{\text{rand}}$.

We now define $x^{\text{rand}}$, which is again symmetric, in the sense that $x^{\text{rand}}_1(V_1, V_2) = x^{\text{rand}}_2(V_1, V_2)$ for every $V_1, V_2 \in T^{\text{rand}}$. We therefore only describe $x^{\text{rand}}_1$. When both agents report types in $\{A', B', C', D', E', F', G', H'\}$, the first three coordinates of each agent’s allocation are given by $x^{\text{det}}$ when fed by the corresponding types in $T^{\text{det}}$, and the remaining coordinates are filled in according to the opponent’s report. More specif-
ically,
\[ \forall V_1 \in \{A', B', C', D'\}, \]
\[ x_1^{\text{rand}}(V_1, A') = [x_1^{\text{det}}(\det(V_1), A), 0, 100, 100, 100], \]
\[ x_1^{\text{rand}}(V_1, B') = [x_1^{\text{det}}(\det(V_1), B), 100, 0, 100, 100], \]
\[ x_1^{\text{rand}}(V_1, C') = [x_1^{\text{det}}(\det(V_1), C), 100, 100, 0, 100], \]
\[ x_1^{\text{rand}}(V_1, D') = [x_1^{\text{det}}(\det(V_1), D), 100, 100, 100, 0]. \] (4)

For the other types, we have
\[ \forall V_1 \in \{E', F', C', H'\}, \forall V_2 \in \{A', B', C', D'\}, \]
\[ x_1^{\text{rand}}(V_1, V_2) = x_2^{\text{rand}}(C', V_2). \]
\[ \forall V_1 \in \{E', F', G', H'\}, \forall V_1 \in T^{\text{rand}}, \]
\[ x_1^{\text{rand}}(V_1, V_2) = [0, 0, 0, 100, 100, 100]. \]

Note that \( x^{\text{rand}} \) itself is deterministic. The difficulty we need to overcome in this section is that the extension of \( x^{\text{rand}} \) may be randomized, and we must show that any extension to \( \text{Conv}(T^{\text{rand}}) \) cannot be DSIC.

The set of feasible allocations is \( \mathcal{F} = \{ x^{\text{rand}}(V_1, V_2) \}_{V_1, V_2 \in T^{\text{rand}}} \). Again we emphasize that the set of feasible allocations is not a product space.

We first show that a subset of the payments are still “locked” as they were in the deterministic setting.

**Lemma 4.** For any allocation rule \( \tilde{x} \) that is an extension of \( x^{\text{rand}} \) if \( \tilde{x} \) can be implemented by a DSIC mechanism with payment rule \( p \), then for any \( V \in \{A', B', C', D'\} \) and any \( V', V'' \in T^{\text{rand}} \), \( p_1(V', V) = p_1(V'', V) \), and \( p_2(V, V') = p_2(V, V'') \).

**Proof.** The proof for types in \( \{A', B', C', D'\} \) follows the same steps as Lemma 1. For any type in \( \{E', F', G', H'\} \), note that they receive the same allocation as type \( C' \) and therefore if they are charged a payment that is different from the payment \( C' \) is charged, then either \( C' \) or that type would be incentivized to deviate.

In order to have a DSIC convex extension in this setting we must satisfy a condition similar to Lemma 2 with the higher dimensional versions of the types from \( T^{\text{rand}} \).

**Lemma 5.** If \( \tilde{x} \) is a DSIC extension of \( x^{\text{rand}} \) then for \( t_1 = \frac{1}{3} A' + \frac{1}{3} B' + \frac{1}{3} D' \), and any \( V_2 \in \{A', B', C', D'\} \), \( \tilde{x}_1(t_1, V_2) \) is supported on \( \{ \tilde{x}_1(A', V_2), \tilde{x}_1(B', V_2), \tilde{x}_1(C', V_2), \tilde{x}_1(D', V_2) \} \).

**Proof.** For the sake of contradiction, assume \( \tilde{x} \) is a DSIC extension of \( x^{\text{rand}} \) implementable by payment rule \( p \), and for some \( V_2 \in \{A', B', C', D'\} \),

there exist a pair of types \( V_1, V_2 \) such that
\[ \Pr[\tilde{x}(t_1, V_2) = x_1^{\text{rand}}(V_1, V_2)] > 0 \] for a \( V_2 \neq V_2 \). Let \( X_{V_2} \) denote the set of allocations \( \{ \tilde{x}_1(A', V_2), \tilde{x}_1(B', V_2), \tilde{x}_1(C', V_2), \tilde{x}_1(D', V_2) \} \).

Let \( \alpha = \Pr[\tilde{x}_1(t_1, V_2) \in X_{V_2}] \) and \( \beta = \Pr[\tilde{x}_1(t_1, V_2) \notin X_{V_2}] \). Then by assumption, \( \beta > 0 \).

Also let \( \tilde{x}_2'(t_1) = \mathbb{E}[\tilde{x}_1(t_1, V_2) | \tilde{x}_1(t_1, V_2) \in X_{V_2}] \) and \( \tilde{x}_2''(t_1) = \mathbb{E}[\tilde{x}_1(t_1, V_2) | \tilde{x}_1(t_1, V_2) \notin X_{V_2}] \). Then the expected value for agent 1 truthfully reporting \( t_1 \) is
\[ \langle t_1, \alpha \tilde{x}_2'(t_1) + \beta \tilde{x}_2''(t_1) \rangle. \]

In order for \( \tilde{x} \) to be DSIC, it must provide type \( t_1 \) with at least as much utility as it could receive from deviating to any type in \( T^{\text{rand}} \).

Let \( v_{\text{max}} \in \arg\max_{V' \in T^{\text{rand}}}(t_1, x_1^{\text{rand}}(V_1', V_2)) \) then \( \tilde{x} \) must satisfy the following DSIC constraint,
\[ \alpha(t_1, \tilde{x}_2'(t_1)) + \beta(t_1, \tilde{x}_2''(t_1)) - p_1(t_1, V_2) \geq \langle t_1, x_1^{\text{rand}}(V_{\text{max}}, V_2) \rangle - p_1(V_{\text{max}}, V_2). \] (5)

We use this constraint to place an upper bound on \( p_1(t_1, V_2) \).

By definition of \( v_{\text{max}} \),
\[ \langle t_1, x_1^{\text{rand}}(V_{\text{max}}, V_2) \rangle \geq \langle t_1, \tilde{x}_2'(t_1) \rangle. \]

Subtracting \( \langle t_1, x_1^{\text{rand}}(V_{\text{max}}, V_2) \rangle \) from both sides of eq. (5) and utilizing the fact that \( \alpha + \beta = 1 \) we have
\[ \beta(t_1, \tilde{x}_2'(t_1)) - \beta(t_1, x_1^{\text{rand}}(V_{\text{max}}, V_2)) \geq p_1(t_1, V_2) - p_1(V_{\text{max}}, V_2). \]

Now using the fact that \( t_1 \)'s maximum difference in value for any two allocations in \( \mathcal{F} \) is upper bounded by \( 3 \), we get
\[ 3\beta + p_1(V_{\text{max}}, V_2) \geq p_1(t_1, V_2). \] (6)

By construction, for \( V_2 \in \{A', B', C', D'\} \), there exists a type \( V_1 \in \{E', F', G', H'\} \) which has value 0 for every allocation in \( X_{V_2} \). (For example if \( V_2 = A' \) then all of the allocations in \( \{ \tilde{x}_1(A', A'), \tilde{x}_1(B', A'), \tilde{x}_1(C', A'), \tilde{x}_1(D', A') \} \) have a 0 in the 4th coordinate and in this case \( v_{\text{not}} = E' \).)

Now by Lemma 4, \( v_{\text{not}} \) receives utility \( -p_1(V_{\text{max}}, V_2) = -p_1(V_{\text{max}}, V_2) \) when reporting truthfully against another type \( V_2 \).

Therefore, in order to keep \( v_{\text{not}} \) from deviating to type \( t_1 \), we must satisfy the following DSIC constraint,
\[ -p_1(V_{\text{max}}, V_2) \geq \beta(v_{\text{not}}, \tilde{x}_2'(t_1)) + \beta(v_{\text{not}}, \tilde{x}_2''(t_1)) \]
\[ -p_1(t_1, V_2). \] (7)

Since \( \tilde{x}_2'(t_1) \) is the expected allocation conditioned on the resulting allocation being an element of \( X_{V_2} \), and \( v_{\text{not}} \) has zero value for any allocation in \( X_{V_2} \), we have \( \langle v_{\text{not}}, \tilde{x}_2'(t_1) \rangle = 0 \).

Therefore,
\[ p_1(t_1, V_2) \geq \beta(v_{\text{not}}, \tilde{x}_2'(t_1)) + p_1(V_{\text{max}}, V_2) \] (8)

Notice that for every allocation outside of \( X_{V_2} \) the type \( v_{\text{not}} \) has value exactly 100 and therefore \( \langle v_{\text{not}}, \tilde{x}_2'(t_1) \rangle = 100 \).

We now use this fact along with eq. (6) and eq. (8) and derive the following contradiction.
\[ 3\beta + p_1(V_{\text{max}}, V_2) - p_1(V_{\text{max}}, V_2) \geq \beta(v_{\text{not}}, x_\beta), \]
\[ 3\beta \geq 100\beta. \]

Since by assumption \( \beta > 0 \), this is a contradiction.

\( \Box \)

We are now ready to prove Theorem 3.
**Proof of Theorem 3.** Consider \( t_1 = \frac{1}{3} A' + \frac{1}{3} B' + \frac{1}{3} D' \) and assume \( \Pr[x_1(t_1, A')] = x_{1\text{rand}}(A', A') < 1 \).

We know from Lemma 5 that \( x_1(t_1, A') \) is supported on \( \{ x_1(A', A'), x_1(B', A'), x_1(C', A'), x_1(D', A') \} \). Since \( x_{1\text{rand}}(A', A') \) is the only allocation for which both \( A' \) and \( B' \) have positive value, \( t_1 \) satisfies
\[
\forall V' \neq A', \quad \langle t_1, x_{1\text{rand}}(V', A') \rangle < \langle t_1, x_{1\text{rand}}(A', A') \rangle.
\]
But in order for \( t_1 \) to have no incentive to report \( A' \) when agent 2's type is \( A' \), we have
\[
\mathbb{E} \left[ \langle t_1, x_1(t_1, A') \rangle \right] - p_1(t_1, A') \geq \langle t_1, x_{1\text{rand}}(A', A') \rangle - p_1(A', A').
\]
Using eq. (9), we have
\[
p_1(A', A') > p_1(t_1, A'). \tag{10}
\]
We now show that, similar to Lemma 5 and via eq. (10), a strictly lower payment for \( t_1 \) as in Equation (10) would violate DSIC constraints. The DSIC constraint that keeps type \( D' \) from deviating to type \( A' \) when the opponent type is \( A' \) is
\[
\langle D', x_{1\text{rand}}(D', A') \rangle - p_1(D', A') \geq \langle D', x_1(t_1, A') \rangle - p_1(t_1, A'). 
\]
Since \( x_1(t_1, A') \) is supported on \( \{ x_1(A', A'), x_1(B', A'), x_1(C', A'), x_1(D', A') \} \) and type \( D' \) values all of these allocations equally we have
\[
\langle D', x_1(t_1, A') \rangle = \langle D', x_{1\text{rand}}(D', A') \rangle,
\]
and by Lemma 4 we have \( p_1(D', A') = p_1(A', A') \). Using these facts we write Equation (11) as
\[
p_1(t_1, A') \geq p_1(A', A').
\]
This contradicts Equation (10).

**Conclusion and Future Work.** We presented a DSIC mechanism on a discrete type space which cannot be implemented if DSIC is required on the convex hull of its type space. Extension from discrete to convex domains is a common step in automated mechanism design, and it is unfortunate that, as an implication of our result, there exists no general procedure for extending an arbitrary DSIC mechanism to the convex hull of its domain. However, as we showed, such extensions are possible in specific settings, with special type spaces (e.g. single dimensional) or special feasible set of allocations (e.g. downward closed or, more generally, single swap feasible). Other sufficient conditions for extensibility should be used to have.
Another possibility we leave open is conditions on an allocation rule (rather than the type space or the set of feasible allocations) that would guarantee its extensibility. We know that an allocation rule is DSIC implementable if and only if it is cyclic monotone. Is there a condition stronger than cyclic monotonicity which guarantees both implementability and extensibility?

The mechanism we constructed for the impossibility result is intricate. A third question we leave open is whether one may find a succinct property that is necessary for extensibility; in other words, is there a natural condition whose violation must result in inextensible mechanisms?

References

[Abebe et al. 2019] Abebe, R.; Cole, R.; Gkatzelis, V.; and Hartline, J. D. 2019. A truthful cardinal mechanism for one-sided matching. CoRR abs/1903.07797.

[Archer and Kleinberg 2014] Archer, A., and Kleinberg, R. 2014. Truthful germs are contagious: A local-to-global characterization of truthfulness. Games and Economic Behavior 86:340–366.

[Ashlagi, Dobzinski, and Lavi 2012] Ashlagi, I.; Dobzinski, S.; and Lavi, R. 2012. Optimal lower bounds for anonymous scheduling mechanisms. Mathematics of Operations Research 37(2):244–258.

[Balcan, Sandholm, and Vitercik 2018] Balcan, M.; Sandholm, T.; and Vitercik, E. 2018. A general theory of sample complexity for multi-item profit maximization. In Proceedings of the 2018 ACM Conference on Economics and Computation, EC 2018 Ithaca, NY, USA, June 18-22, 2018, 173–174.

[Chawla et al. 2010] Chawla, S.; Hartline, J. D.; Malec, D. L.; and Sivan, B. 2010. Multi-parameter mechanism design and sequential posted pricing. In Proceedings of the 42nd ACM Symposium on Theory of Computing, STOC 2010, Cambridge, Massachusetts, USA, 5–8 June 2010, 311–320.

[Chawla et al. 2013] Chawla, S.; Hartline, J. D.; Malec, D. L.; and Sivan, B. 2013. Prior-independent mechanisms for scheduling. In Symposium on Theory of Computing Conference, STOC 2013, Palo Alto, CA, USA, June 1-4, 2013, 51–60.

[Chawla, Malec, and Sivan 2010] Chawla, S.; Malec, D. L.; and Sivan, B. 2010. The power of randomness in bayesian optimal mechanism design. In Proceedings 11th ACM Conference on Electronic Commerce, EC 2010, Cambridge, Massachusetts, USA, June 7-11, 2010, 149–158.

[Clarke 1971] Clarke, E. 1971. Multimart pricing of public goods. Public Choice 11(1):17–33.

[Conitzer and Sandholm 2002] Conitzer, V., and Sandholm, T. 2002. Complexity of mechanism design. In Proceedings of the 18th Conference in Uncertainty in Artificial Intelligence,UAI 2002, University of Alberta, Edmonton, Alberta, Canada, August 1-4, 2002, 103–110.

[Daskalakis, Deckelbaum, and Tzamos 2017] Daskalakis, C.; Deckelbaum, A.; and Tzamos, C. 2017. Strong duality for a multiple-good monopolist. Econometrica 85(3):735–767.

[Devanur, Mihail, and Vazirani 2005] Devanur, N. R.; Mihail, M.; and Vazirani, V. V. 2005. Strategyproof cost-sharing mechanisms for set cover and facility location games. Decision Support Systems 39(1):11–22.

[Dobzinski, Fu, and Kleinberg 2015] Dobzinski, S.; Fu, H.; and Kleinberg, R. 2015. Approximately optimal auctions for correlated bidders. Games and Economic Behavior 92:349–369.

[Feldman, Gravin, and Lucier 2015] Feldman, M.; Gravin, N., and Lucier, B. 2015. Combinatorial auctions via posted prices. In Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2015, San Diego, CA, USA, January 4-6, 2015, 123–135.

[Feng and Hartline 2018] Feng, Y., and Hartline, J. D. 2018. An end-to-end argument in mechanism design (prior-independent auctions for budgeted agents). In 59th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2018, Paris, France, October 7-9, 2018, 404–415.

[Groves 1973] Groves, T. 1973. Incentives in teams. Econometrica 41(4):617–31.

[Guo and Conitzer 2010] Guo, M., and Conitzer, V. 2010. Computationally feasible automated mechanism design: General approach and case studies. In Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2010, Atlanta, Georgia, USA, July 11-15, 2010.

[Likhodedov and Sandholm 2004] Likhodedov, A., and Sandholm, T. 2004. Methods for boosting revenue in combinatorial auctions. In Proceedings of the Nineteenth National Conference on Artificial Intelligence, Sixteenth Conference on Innovative Applications of Artificial Intelligence, AAAI 2004 July 25-29, 2004, San Jose, California, USA, 232–237.

[Likhodedov and Sandholm 2005] Likhodedov, A., and Sandholm, T. 2005. Approximating revenue-maximizing combinatorial auctions. In Proceedings, The Twentieth National Conference on Artificial Intelligence and the Seventeenth Innovative Applications of Artificial Intelligence Conference,AAAI 2005, July 9-13, 2005, Pittsburgh, Pennsylvania, USA, 267–274.

[Morgenstern and Roughgarden 2016] Morgenstern, J., and Roughgarden, T. 2016. Learning simple auctions. In Proceedings of the 29th Conference on Learning Theory, COLT 2016, New York, USA, June 23-26, 2016, 1298–1318.

[Myerson 1981] Myerson, R. B. 1981. Optimal auction design. Mathematics of Operations Research 6(1):58–73.

[Nisan and Ronen 2001] Nisan, N., and Ronen, A. 2001. Algorithmic mechanism design. Games and Economic behavior 35(1-2):166–196.

[Rochet 1987] Rochet, J.-C. 1987. A necessary and sufficient condition for rationalizability in a quasi-linear context. Journal of Mathematical Economics 16(2):191 – 200.

[Saks and Yu 2005] Saks, M. E., and Yu, L. 2005. Weak monotonicity suffices for truthfulness on convex domains. In
Supplementary Material

Proof of Theorem 1

Proof. For ease of notation let \( \overline{0} \) represent the “zero-allocation” (i.e. an all-zeroes vector of length \( m \)). We now give, for any DSIC and IR \((x,p)\), an extended allocation rule \( \tilde{x} \) and extended payment rule \( \tilde{p} \) that are DSIC and IR.

Whenever the reported type profile \( t \) contains only types from the original typespace \( T \) (i.e. \( t \in T \)) then by definition our extended allocation rule is \( \tilde{x}(t) = x(t) \) and our extended payment rule is \( \tilde{p}(t) = p(t) \).

If exactly one agent \( i \) has a type \( t_i \in \text{Conv}(T_i) \setminus T_i \) and their opponents have types \( t_{-i} \in T_{-i} \), then agent \( i \)'s allocation is the same as allocation as either the type \( t_i^{\text{max}} = \text{argmax}_{t_i' \in T_i} (t_i, x_i(t_i', t_{-i})) - p_i(t_i', t_{-i}) \) or the “zero-allocation”, whichever provides more utility. More formally, if \( t_i \in \text{Conv}(T_i) \setminus T_i \) and \( t_{-i} \in T_{-i} \), the extended allocation rule for \( t_i \) is

\[
\tilde{x}_i(t_i, t_{-i}) = \begin{cases} 
    x_i(t_i^{\text{max}}, t_{-i}), & \text{if } (t_i, x_i(t_i^{\text{max}}, t_{-i})) - p_i(t_i^{\text{max}}, t_{-i}) \geq 0; \\
    \overline{0}, & \text{otherwise;}
\end{cases}
\]

and the extended payment rule is

\[
\tilde{p}_i(t_i, t_{-i}) = \begin{cases} 
    p_i(t_i^{\text{max}}, t_{-i}), & \text{if } (t_i, x_i(t_i^{\text{max}}, t_{-i})) - p_i(t_i^{\text{max}}, t_{-i}) \geq 0; \\
    0, & \text{otherwise.}
\end{cases}
\]

All other agents \( j \neq i \) are given the “zero-allocation” and charged a payment of 0 (i.e. \( \tilde{x}_j(t_j, t_{-j}) = \overline{0} \) and \( \tilde{p}_j(t_j, t_{-j}) = 0 \)).

The allocation is feasible since \( x(t_i^{\text{max}}, t_{-i}) \in \mathcal{F} \) and \( \mathcal{F} \) is downward closed.

Finally if more than one agent has a type which is not in the original typespace then \( \forall i, \tilde{x}_i(t_i, t_{-i}) = \overline{0} \) and \( \tilde{p}_i(t_i, t_{-i}) = 0 \).

It remains to be shown that \( \tilde{x} \) and \( \tilde{p} \) are DSIC and IR. To see that \( \tilde{x} \) and \( \tilde{p} \) are IR observe that every “new” allocation and payment that are not zero are only chosen if they provide non-negative utility and since the “zero-allocation” with a payment of 0 by definition provides 0 utility, \( \tilde{x} \) must satisfy IR constraints.

We now show that \( \tilde{x} \) and \( \tilde{p} \) satisfy DSIC constraints. For any agent \( i \) if they have an opponent \( j \) who reports \( t_j \in \text{Conv}(T_j) \setminus T_j \) then agent \( i \) is given the zero-allocation, regardless of their reported type. Therefore, we trivially satisfy DSIC constraints whenever an opponent type profile contains a “new” type.

When \( t_{-i} \in T_{-i} \) we must show that agent \( i \) has no incentive to deviate, both when they have a “old” type and when they have an “new” type. For an “old” type \( t_i \in T_i \), deviating to some type \( t_i' \in \text{Conv}(T_i) \setminus T_i \) gives either the same utility as deviating to \( t_i^{\text{max}} = \text{argmax}_{t_i' \in T_i} (t_i', x_i(t_i', t_{-i})) - p_i(t_i', t_{-i}) \) or utility 0. Since \( (x,p) \) is DSIC and IR, this deviation cannot provide higher utility than \( (t_i, \tilde{x}_i(t_i, t_{-i})), \tilde{p}_i(t_i, t_{-i}) \). Finally, for a “new” type \( t_i \in \text{Conv}(T_i) \setminus T_i \), by definition of \( \tilde{x} \), the agent by reporting truthfully is given the allocation and payment that maximizes utility over all possible deviations in \( T_i \); by the logic above, this maximizes utility over all possible deviations in \( \text{Conv}(T_i) \) as well. Therefore, the agent has no incentive to deviate.

Proof of Theorem 2

Proof. This proof follows the same steps as the proof of Theorem 1 with swap feasible allocations now playing the role that “zero-allocations” played in Theorem 1. Recall that a swap allocation for agent \( i \), denoted \( x_i^{\text{ssf}}(t_i) \), is a feasible allocation vector with the property that if we replace the \( i \)th component of this vector with the \( i \)th component from any other feasible allocation, the resulting allocation is still feasible.

For any DSIC \((x,p)\) on \( T \), We give a DSIC extension \((\tilde{x}, \tilde{p})\) on \( \text{Conv}(T) \).

Whenever the reported type profile \( t \) contains only types from the original typespace \( T \) (i.e. \( t \in T \)), \( \tilde{x}(t) = x(t) \) and \( \tilde{p}(t) = p(t) \), as required by definition of an extension.

If only one agent \( i \) has a type \( t_i \in \text{Conv}(T_i) \setminus T_i \) and their opponents have types \( t_{-i} \in T_{-i} \), then agent \( i \) is given the same allocation as the type \( t_i^{\text{max}} = \text{argmax}_{t_i' \in T_i} (t_i, x_i(t_i', t_{-i})) - p_i(t_i', t_{-i}) \). More formally, for \( t_i \in \text{Conv}(T_i) \setminus T_i \) and \( t_{-i} \in T_{-i} \), the extended allocation rule for \( t_i \) is \( \tilde{x}_i(t_i, t_{-i}) = x_i(t_i^{\text{max}}, t_{-i}) \) and the extended payment rule is \( \tilde{p}_i(t_i, t_{-i}) = p_i(t_i^{\text{max}}, t_{-i}) \).

The allocations for the remaining agents are described by agent \( i \)'s swap feasible allocation, and are charged a payment of 0. Formally, \( \forall j \neq i, \tilde{x}_j(t_j, t_{-j}) = x_i^{\text{ssf}}(t_j) \) and \( \tilde{p}_j(t_j, t_{-j}) = 0 \).

By definition of single swap feasibility, regardless of the value of \( \tilde{x}_i(t_i, t_{-i}), \tilde{x}_j(t_j, t_{-j}) \), \( x_i^{\text{ssf}}(t_i) \) remains feasible.

Finally, if a (reported) type profile \( t \) has more than one agent with a type not in the original typespace, among agents whose types are not within their original typespaces, let \( j \) and \( k \) be the smallest and second smallest indices among them. The extended allocation for such a type profile \( t \) is \( \tilde{x}(t) = (x_j^{\text{ssf}}(k), x_k^{\text{ssf}}(j)) \) and \( \forall i, \tilde{p}_i(t) = 0 \). In other words, every agent’s allocation is determined by the swap allocation of agent \( j \), except agent \( j \) himself, whose allocation is given by the swap allocation of agent \( k \). Since this allocation only changes the \( j \)th component of \( x_i^{\text{ssf}}(t) \), it is feasible by definition of SSF.

It remains to show that \((\tilde{x}, \tilde{p})\) is DSIC. First we deal with the case where an agent \( i \) has at least one opponent whose type is not in the original type space. Let \( j \) be the smallest index of such an opponent. Then agent \( i \) is given \( x_i^{\text{ssf}}(j) \) regardless of their reported type. Therefore we trivially satisfy DSIC when an opponent type profile contains a “new” type.
When $t_{-i} \in T_{-i}$, we must show that any agent $i$ has no incentive to deviate, both when they have an “old” type and when they have an “new” type. For an “old” type $t_i \in T_i$, deviating to some type $t'_i \in \text{Conv}(T_i) \setminus T_i$ gives the same utility as deviating to $t^\max_i = \arg\max_{t''_i \in T_i} \langle t_i, x_i(t''_i, t_{-i}) \rangle - p_i(t''_i, t_{-i})$. Since $(x, p)$ is DSIC, such a deviation cannot provide higher utility than $\langle t_i, x_i(t_i, t_{-i}) \rangle - \tilde{p}_i(t_i, t_{-i})$. Finally, if $t_i \in \text{Conv}(T_i) \setminus T_i$, then by definition of $\tilde{x}$, by reporting truthfully the agent is given the allocation and payment that maximizes their utility over all possible deviations in $T_i$; by the logic above, this maximizes utility over all possible deviations in $\text{Conv}(T_i)$ as well. Therefore, the agent has no incentive to deviate.
Figure 1: The maximum value for allocations in $x_{1}^{\text{det}}$ for all types in $\text{Conv}(T_{1}^{\text{det}})$. Each plot (visually an upside-down hollow pyramid) represents a reported type for agent 2. From left to right these represent agent 2’s type being $A$, $B$ or $C$ ($t_{2} = D$ is omitted). Within each plot, or set of planes, the color represents the type in $T_{1}^{\text{det}}$ agent 1 should report to receive the highest value. The green plane represents $t_{1} = A$, the blue plane represents $t_{1} = B$ and the red plane represents $t_{1} = C$. For example, the green plane in the first set of planes represents the value from allocation $x_{1}^{\text{det}}(A, A)$. From this image we can gain the intuition behind the “locked” payments by re-imagining these planes as describing utility when payments are zero. We can think about changes in payments as vertical shifts in the planes. For example, changing $p_{1}^{\text{det}}(A, A)$ would shift the plane corresponding to $x_{1}^{\text{det}}(A, A)$. This would either cause it to fall below the red plane at the point $A$ giving $A$ incentive to deviate to type $B$ or cause it to rise above the blue plane at point $B$ giving $B$ an incentive to deviate to type $A$.

Figure 2: The typespace $\text{Conv}(T^{\text{det}})$ projected to a 2 dimensional plane. The types in $T^{\text{det}}$ are labeled.