# Exponential Communication Separations between Notions of Selfishness

Aviad Rubinstein*
Stanford University
U.S.A.

Raghuvansh R. Saxena†
Princeton University
U.S.A.

Clayton Thomas‡
Princeton University
U.S.A.

S. Matthew Weinberg§
Princeton University
U.S.A.

Junyao Zhao¶
Stanford University
U.S.A.

## ABSTRACT

We consider the problem of implementing a fixed social choice function between multiple players (which takes as input a type $t_i$ from each player $i$ and outputs an outcome $f(t_1, \ldots, t_n)$), in which each player must be incentivized to follow the protocol. In particular, we study the communication requirements of a protocol which: (a) implements $f$, (b) implements $f$ and computes payments that make it ex-post incentive compatible (EPIC) to follow the protocol, and (c) implements $f$ and computes payments in a way that makes it dominant-strategy incentive compatible (DSIC) to follow the protocol.

We show exponential separations between all three of these quantities, already for just two players. That is, we first construct an $f$ such that $f$ can be implemented in communication $c$, but any EPIC implementation of $f$ (with any choice of payments) requires communication $\exp(c)$. This answers an open question of [Fadel and Segal, 2009; Babaioff et. al., 2013]. Second, we construct an $f$ such that an EPIC protocol implements $f$ with communication $C$, but all DSIC implementations of $f$ require communication $\exp(C)$.

## CCS CONCEPTS

• **Theory of computation → Solution concepts in game theory**; **Algorithmic mechanism design**.

## KEYWORDS

Implementation theory, Algorithmic mechanism design, Communication complexity

*Supported by NSF CCF- 1954927, and a David and Lucile Packard Fellowship.
†Supported by the National Science Foundation CAREER award CCF-1750443 and a Microsoft PhD Fellowship.
‡Supported by NSF-CCF 1955205.
§Supported by NSF-CCF 1955205.
¶Supported by NSF CCF-1954927.


*STOC '21, June 21–25, 2021, Virtual, Italy*

ACM ISBN 978-1-4503-8053-9/21/06…$15.00
https://doi.org/10.1145/3406325.3451127

**ACM Reference Format:**
Aviad Rubinstein, Raghuvansh R. Saxena, Clayton Thomas, S. Matthew Weinberg, and Junyao Zhao. 2021. Exponential Communication Separations between Notions of Selfishness. In *Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing (STOC '21), June 21–25, 2021, Virtual, Italy*. ACM, New York, NY, USA, 14 pages. https://doi.org/10.1145/3406325.3451127

## 1 INTRODUCTION

Consider the following canonical problem: there is a set $Y$ of possible outcomes, and each of $n$ players have a type $t_i$ which determines their utility $u_i(t_i, y)$ for each outcome $y \in Y$. You have a particular *social choice function* $f$ in mind, which maps a profile of types $\vec{t} = (t_1, \ldots, t_n)$ to $f(\vec{t}) \in Y$. A canonical question within Computer Science might first ask "what is $CC(f)$, the communication complexity of $f$?" That is, over all deterministic protocols computing $f$ among the $n$ players (who initially each know only their own type, and not that of any others), which one uses the least number of bits in the worst case?

But consider now the possibility that the players do not simply follow the intended protocol, and instead strive to maximize their own utility. The need to incentivize the players to follow the protocol motivates the entire field of Algorithmic Mechanism Design, as well as questions such as "*what is the communication complexity to implement $f$, using a protocol which incentivizes the players to follow it?*"

There are several formal instantiations of this question, depending on how strongly one wishes to incentivize the players. One common solution concept is ex-post incentive compatibility (EPIC), where the protocol may charge prices and it is in each player's interest to follow the protocol assuming that other players follow the protocol as well (formally, it is a Nash equilibrium to follow the protocol, no matter the other players' types). We let $CC^{EPIC}(f)$ denote the minimum communication cost of an EPIC protocol implementing $f$. Another common solution concept is dominant strategy incentive compatibility (DSIC), where the protocol may charge prices and it is in each player's interest to follow the protocol no matter what the other players do (even if that behavior is completely irrational). We let $CC^{DSIC}(f)$ denote the minimum communication cost of a DSIC protocol implementing $f$. Because any EPIC protocol must compute $f$ and any DSIC protocol is in particular an EPIC protocol, we have $CC(f) \le CC^{EPIC}(f) \le CC^{DSIC}(f)$.

Formally, we study the following question: *for a fixed $f$, how does $CC(f)$ relate to $CC^{EPIC}(f)$, and how does $CC^{EPIC}(f)$ relate*

*to* $CC^{DSIC}(f)$? While related directions have received substantial attention and produced a vast body of works (we overview this related work, and others, in Section 1.1), relatively little attention has been paid to these fundamental questions. Our main results are exponential separations between all three quantities (and these are the first such separations). Specifically:

Theorem (See Theorem 3.1 and Theorem 4.10). *There exists* $f$ *such that* $CC^{EPIC}(f) = \exp(CC(f))$. *There exists* $g$ *such that* $CC^{DSIC}(g) = \exp(CC^{EPIC}(g))$.

The gap in both cases is at most exponential, so this is the largest gap possible.[1]

## 1.1 Context and Related Work

There is a *vast* literature studying the communication requirements of protocols for honest players versus truthful mechanisms for strategic players [2, 3, 7–13, 15–17, 19, 20, 22–24, 26, 27]. Our work certainly fits into this literature, but goes in a fairly distinct direction. Specifically, this literature nearly-ubiquitously considers comparisons between how much communication is required for *some* $f$ satisfying some property (e.g. guaranteeing an $\alpha$-approximation to the optimal welfare[2]) versus how much communication is required for *some* EPIC implementation of *some* $g$ guaranteeing that property. In particular, $f$ and $g$ may be different social choice functions, and separations normally arise because the lowest-communication $f$ guaranteeing the desired property *has no EPIC implementation* — there simply don't exist prices that make any implementation of $f$ EPIC, no matter how much communication is used.[3]

Our work studies a fundamentally different question: for a fixed $f$ *which is EPIC-implementable*, how much communication overhead is required to actually compute prices which make the implementation EPIC? For an example of this distinction, consider a single-item auction: each player has a value $v_i$ for the item. The space of outcomes can award the item to any bidder, or no one. The social choice function $f$ which gives the item to the highest bidder can be EPIC-implemented (by the second-price auction). The social choice function $g$ which gives the item to the lowest bidder cannot be EPIC-implemented (by any prices, no matter how much communication). In general, many approximation algorithms for richer settings tend to be like $g$: they are simply not implementable, no matter what. So the driving force behind all prior work is separating the approximation guarantees for efficient protocols which are EPIC-implementable (and tend to have low overhead to actually compute the prices), versus those which are not.

There is significantly less prior work addressing our specific questions. The direction was first posed in [18], who explicitly pose the question of $CC(f)$ versus $CC^{EPIC}(f)$, and demonstrate that $CC^{EPIC}(f)$ can be strictly larger than $CC(f)$. Follow-up work of [4] were the first to make progress on this, and show a separation of $CC(f)$ versus $CC^{EPIC}(f)$ which is linear in the number of players (so in particular, the blow-up for two players is not large). In comparison to these works, our Theorem 3.1 shows the maximum possible gap (exponential) with just two players, resolving the open question in [18].

[18, Appendix B.2] defines and discusses $CC^{DSIC}$, but only considers the relationship between $CC$ and $CC^{DSIC}$ (not the gap between $CC^{EPIC}$ and $CC^{DSIC}$). [11, Appendix C.1] shows that no large separation between $CC^{EPIC}(f)$ and $CC^{DSIC}(f)$ is possible for the particular setting of two player combinatorial auctions with arbitrary monotone valuations[4].

The study of $CC^{EPIC}(f)$ versus $CC^{DSIC}(f)$ is conceptually related to a recent push with the Economics and Computation community to understand obviously strategyproof (OSP) mechanisms [1, 6, 25, 28]. These works do not focus on communication complexity, but rather on characterizing implementations which satisfy OSP (a stronger, but related, definition than DSIC). In comparison to these works, our Theorem 4.10 bears technical similarity, and our approach may be useful for proving communication lower bounds on OSP implementations.

[18] also study related questions for a solution concept termed "Bayesian incentive compatibility" (BIC), and they obtain a tight exponential separation of $CC$ and $CC^{BIC}$. [5] studies the solution concept termed "truthful in expectation" (TIE), and show that in *single-parameter* settings there is no (substantial) separation between $CC(f)$ and $CC^{TIE}(f)$.

**Concurrent and Independent Work.** Concurrently and independently of our work, Dobzinski and Ron [14] also consider the relationship between $CC$ and $CC^{EPIC}$.[5] In particular, they also provide a construction of a function $f$ witnessing $CC^{EPIC}(f) = \exp(CC(f))$ (their Section 3.1), which is similar to ours (our Section 3) in that it derives hardness from high-precision prices. The remainder of their paper is disjoint from ours (in particular, they do not study $CC^{DSIC}$, so there is no analogue to our Section 4). Instead, they establish the following results: (a) There exist functions $f$ with $CC^{EPIC}(f) = \exp(CC(f))$ *without* high-precision prices (but with a third bidder). (b) Under certain assumptions on $f$, $CC^{EPIC}(f) = \text{poly}(n, CC(f))$ and/or $CC^{TIE}(f) = \text{poly}(n, CC(f))$. (c) Reconstructing the *menu* presented by an EPIC mechanism can be exponentially harder than computing the mechanism alone. A high-level distinction of our works is that our paper provides exponential separations between multiple solution concepts (algorithmic vs. EPIC vs. DSIC), whereas their paper provides a more thorough investigation of algorithmic vs. EPIC.

## 1.2 Summary and Roadmap

We establish an exponential separation between $CC(f)$ and $CC^{EPIC}(f)$, and also $CC^{EPIC}(f)$ and $CC^{DSIC}(f)$, both the largest

[1] To see this, consider the following sketch: for every protocol, there exists a simultaneous protocol (with one round of communication) with at most an exponential blowup in communication. A simultaneous protocol is EPIC if and only if it is DSIC, and [18, Proposition 1] shows that EPIC prices can be added to any simultaneous protocol for $f$ with low overhead. So the gap between the three quantities can be no larger than the gap between simultaneous and interactive communication requirements for $f$, which is at most exponential.

[2] The welfare of an outcome $y$ is defined as $\sum_i u_i(t_i, y)$.

[3] On the other hand, if $f$ is EPIC-implementable, it is DSIC-implementable, but perhaps with exponential overhead.

[4] The proof of [11, Appendix C.1] relies on the fact that incentive compatible combinatorial auctions with arbitrary monotone valuations have low "taxation complexity". Our construction in Section 4 circumvents this theorem because its environment is a very structured subset of two player monotone combinatorial auctions, and moreover, our social choice function $f$ has high taxation complexity.

[5] Both papers were uploaded to arXiv simultaneously on December 29th, 2020.

possible, and first of their kind. Section 3 provides the separation between $CC(f)$ and $CC^{EPIC}(f)$, and Section 4 provides the separation between $CC^{EPIC}(f)$ and $CC^{DSIC}(f)$.

## 2 PRELIMINARIES

We study *implementations* of *social choice functions* over *(social choice) environments*. For completeness and accessibility for the reader not familiar with game theory, we rigorously define all of these terms in Appendix A. Here, we briefly and intuitively describe the central definitions of the paper.

The environment specifies a set of outcomes $Y$ and a set of types $\mathcal{T}_1, \ldots, \mathcal{T}_n$ for the $n$ different strategic agents. Intuitively, the types represent the different possible options for "who each agent might be". When agent $i$ has type $t_i \in \mathcal{T}_i$, they have utility $u_i(t_i, y) \in \mathbb{R}$ for each outcome $y \in Y$. When we study environment with transfers[6], we assume utilities are quasilinear (that is, if outcome $y$ is selected and agent $i$ receives transfer $p$, then agent $i$ gets utility $u_i(t_i, y) + p$). The social choice function $f : \mathcal{T}_1 \times \ldots \times \mathcal{T}_n \to Y$ specifies how the outcome depends on the type each agent has. While the "social planner" designing the mechanism wishes to compute $f$, the agents wish to maximize their own utility. The social choice function itself is assumed to be implementable. That is, there exists transfer functions $p_1, \ldots, p_n : \mathcal{T}_1 \times \cdots \times \mathcal{T}_n \to \mathbb{R}$ for each agent, such that for all $i$, types $t_1, \ldots, t_n$, and $t_i'$, we have

$$f(t_i, t_{-i}) + p_i(t_i, t_{-i}) \geq f(t_i', t_{-i}) + p_i(t_i', t_{-i}).$$

We say that transfers $(p_1, \ldots, p_n)$ *incentivize* $f$, and we say that $f$ is incentive compatible without transfers if each $p_i(\cdot)$ above can be taken to be 0.

A mechanism consists of an (extensive form) game $G$ which the $n$ agents play, and "type-strategies" $S_1, \ldots, S_n$ which suggest how the agents should play $G$. Intuitively, the game $G$ iteratively solicits actions from players, updating its state according to the action chosen, and outputting some result after a finite amount of time. This is represented by a game tree, where the nodes correspond to states of the game. Each non-leaf node is labeled by some agent, and the edges below that node are labeled with the actions that agent may play at that state of the game. The game is not perfect information: it may hide information from agents or ask them to act simultaneously. For each agent $i \in [n]$, the states of $G$ at which $i$ is called to act are partitioned into "information sets" $I_i \in \mathcal{I}_i$, where two nodes are in the same information set if and only if agent $i$ cannot distinguish between them while playing the game[7]. For $i \in [n]$, the type-strategy $S_i$ maps types $t_i \in \mathcal{T}_i$ to "behavioural strategies" $s_i = S_i(t_i)$ which player $i$ can play in $G$. A behavioural strategy (typically referred to simply as a strategy) specifies the action that player $i$ will choose any time they are called to act over the course of the game, that is, it assigns an action to each information set of player $i$. We denote the result output by $G$ when the agents play strategies $s_1, \ldots, s_n$ by $G(s_1, \ldots, s_n)$.

[6]Throughout the paper, we make no assumptions on the transfers. That is, they can be positive or negative, and an agent can receive negative utility. This makes our impossibility results only stronger.

[7]We assume the game satisfies "perfect recall", that is, the game cannot force agents to forget information they knew in the past. For details on how information sets are defined, see Appendix A.

A mechanism $G$ with strategies $S_1, \ldots, S_n$ *computes (without transfers)* a social choice function $f$ if $G(S_1(t_1), \ldots, S_n(t_n)) = f(t_1, \ldots, t_n)$. A mechanism *computes* $f$ *(with transfers)* if the result of the game additionally includes transfers $p_1, \ldots, p_n$ to each player.

We consider two notions of incentive compatibility for interactive mechanisms. In words, a mechanism is dominant strategy incentive compatible (DSIC) if, for *any* (behavioral) strategy profile $s_{-i} := (s_j)_{j \neq i}$ of the other players, it is a best response for player $i$ to play $S_i(t_i)$. That is, for all $t_i, s_{-i}, s_i'$, we have

$$u_i(t_i, G(S_i(t_i), s_{-i})) \geq u_i(t_i, G(s_i', s_{-i})),$$

where we recall that if the mechanism has transfers, $u_i(t_i, \cdot)$ is the quasilinear utility given by agent $i$'s value for the outcome when their type is $t_i$, plus the transfer $p_i$ to player $i$. On the other hand, a mechanism is ex-post Nash incentive compatible (EPIC) if, for any profile of strategy $S_{-i}(t_{-i}) := (S_j(t_j))_{j \neq i}$ which are *consistent with type-strategies* $S_{-i}$, it is a best response to play $S_i(t_i)$. That is, for all $t_i, t_{-i}, s_i'$, we have

$$u_i(t_i, G(S_i(t_i), S_{-i}(t_{-i})) \geq u_i(t_i, G(s_i', S_{-i}(t_{-i}))).$$

Observe quickly the following approach for an EPIC implementation of $f$: Say that $(p_1, \ldots, p_n)$ incentivizes $f$. Then one can run protocols separately to compute $f$, and also to compute each $p_i$, and then output all of these together. This is simply because the EPIC constraints assume that the other bidders' strategies are fixed by their type. So the overhead of $CC^{EPIC}(f)$ versus $CC(f)$ is exactly the overhead to compute transfers. This *does not* hold for DSIC implementations. Indeed, this is because other bidders may use a bizarre (not utility-maximizing) strategy which changes their behavior in (e.g.) the protocol to compute $p_i$ as a function of your behavior in the protocol to compute $f$. But the EPIC condition does not require guarantees against such bizarre strategies, only the fixed strategies which guarantee each player a best response (assuming other players also use such a strategy). We formally define our complexity measures as follows:

Definition 2.1. *For an arbitrary social choice function $f$,*

- *$CC(f)$ is the minimum communication cost of a mechanism (no incentives) computing $f$.*
- *If $f$ is implementable, $CC^{EPIC}(f)$ is the minimum value of $CC(f, p_1, \ldots, p_n)$ over any transfer functions $p_1, \ldots, p_n$ which incentivize $f$.*
- *If $f$ is implementable, $CC^{DSIC}(f, p_1, \ldots, p_n)$ is the minimum communication cost of any DSIC mechanism computing $(f, p_1, \ldots, p_n)$. Moreover, $CC^{DSIC}(f)$ is the minimum value of $CC^{DSIC}(f, p_1, \ldots, p_n)$ for any transfer functions $p_1, \ldots, p_n$ which incentivize $f$.*

## 3 EXPONENTIAL SEPARATION OF $CC(f)$ AND $CC^{EPIC}(f)$

In this section, we show that there exists an implementable social choice function $f$ which has communication complexity $O(\log n)$, yet any EPIC implementation of $f$ must use $\Omega(n)$ communication.

We now describe our construction at a high level. Our instance has two players, Alice and Bob. Alice's type can be represented succinctly, but Bob's type is "complicated". Therefore, without regards to incentives, this social choice function can be efficiently

computed in two rounds, with Alice sending her type to Bob in the first round, and Bob deciding the outcome in the second round. However, the social choice function and the utilities of Alice are designed carefully such that there is essentially only one possible transfer function that gives an EPIC implementation, and moreover, this transfer function has to be "as complicated as the types of Bob". This means that the communication required to EPIC implement the social choice function is large.

*Social choice environment.* Consider a 2-player social choice environment and refer to the players as Alice and Bob. The space of outcomes of the environment is $[n+1]$. The class of Bob's types is $\mathcal{T}_B = \{0,1\}^n$. That is, Bob's type is a binary string $b$ of length $n$. We let $b_i \in \{0,1\}$ denote $b$'s $i$-th coordinate. Bob's utility is always zero regardless of the outcome (that is, $u_B(b,i) = 0$ for all $i \in [n+1], b \in \mathcal{T}_B$). The class of Alice's types is $\mathcal{T}_A = \bigcup_{i\in[n]}\{a_{i,\ell}, a'_{i,\ell}, a_{i,h}, a'_{i,h}\}$, where for each $i \in [n]$, the types $a_{i,\ell}, a'_{i,\ell}, a_{i,h}, a'_{i,h}$ have utility:

$$\begin{aligned} &u_A(a_{i,\ell}, i) = 2^{-n} && u_A(a'_{i,\ell}, i) = 0\\ &u_A(a_{i,\ell}, i+1) = 0 && u_A(a'_{i,\ell}, i+1) = 2^{-n}\\ &u_A(a_{i,h}, i) = 2^{-n} && u_A(a'_{i,h}, i) = 0\\ &u_A(a_{i,h}, i+1) = 2^{i} && u_A(a'_{i,h}, i+1) = 2^{i} + 2^{-n}, \end{aligned}$$

and $u_A(a_{i,\ell}, j) = u_A(a'_{i,\ell}, j) = u_A(a_{i,h}, j) = u_A(a'_{i,h}, j) = -\infty$ for all other outcomes $j \notin \{i, i+1\}$. Intuitively, $a_{i,\ell}, a'_{i,\ell}$ are "low types" of Alice, and $a_{i,h}, a'_{i,h}$ are "high types" (which get much more utility from outcome $i+1$).

*Social choice function.* The social choice function $f : \mathcal{T}_A \times \mathcal{T}_B \to [n+1]$ is given by

$$\begin{aligned} &f(a_{i,\ell}, b) = i && f(a'_{i,\ell}, b) = i + 1 - b_i\\ &f(a_{i,h}, b) = i + 1 - b_i && f(a'_{i,h}, b) = i + 1. \end{aligned}$$

That is, each of Alice's type among $a_{i,\ell}, a'_{i,\ell}, a_{i,h}, a'_{i,h}$ receives either outcome $i$ or $i+1$, and the exact outcome chosen depend on Bob's type $b$ in the following way: If $b_i = 0$, then $a_{i,\ell}$ receives outcome $i$, and each of $a'_{i,\ell}, a_{i,h}, a'_{i,h}$ receives outcome $i+1$. If $b_i = 1$, then each of $a_{i,\ell}, a'_{i,\ell}, a_{i,h}$ receives outcome $i$, and $a'_{i,h}$ receives outcome $i+1$.

Theorem 3.1. *In the 2-player environment above, the social choice function $f$ is EPIC implementable. Moreover, there is an exponential separation between the communication complexity for computing $f$ and the communication complexity of any EPIC implementation of $f$, i.e.,*

$$CC(f) = O(\log n) \qquad CC^{EPIC}(f) = \Theta(n).$$

Proof. First, observe that Alice and Bob can compute $f$ with $O(\log n)$ communication in the following way: Alice sends her valuation, which can be described with $O(\log n)$ bits, to Bob, and then, Bob computes and outputs the outcome, which also costs $O(\log n)$ bits. Thus, $CC(f) = O(\log n)$.

On the other hand, consider any EPIC implementation of $f$. Without loss of generality, we may assume that the transfers to Bob are always 0. Let $p(a,b)$ denote the transfer given to Alice when Alice has type $a \in \mathcal{T}_A$ and Bob has type $b \in \mathcal{T}_B$. By standard arguments, we must have $p(a',b) = p(a,b)$ for any $b \in \mathcal{T}_B$ and $a, a' \in \mathcal{T}_A$ such that $f(a',b) = f(a,b)$ (otherwise, one of $a$ or $a'$ would want to deviate to the other, in order to get a higher transfer for the same outcome). Thus, going forward we write the transfer function $p : [n+1] \times \mathcal{T}_B \to \mathbb{R}$, where $p(i,b)$ is the transfer to Alice when Bob has type $b$ and outcome $i$ is the output of $f$.

Now we prove our main lemma, which allows us to characterize $p$ in any EPIC implementation of $f$.

Lemma 3.2. *Transfers $p$ incentivize $f$ if and only if we have*

$$p(i,b) - p(i+1,b) \in [b_i 2^i - 2^{-n},\ b_i 2^i + 2^{-n}] \tag{*}$$

*for all $i \in [n]$ and $b \in \mathcal{T}_B$.*

Proof. When Alice has type $a_i \in \{a_{i,\ell}, a'_{i,\ell}, a_{i,h}, a'_{i,h}\}$, the social choice function $f$ will select outcome $i$ or $i+1$, based on the type of Alice and bit $b_i$ of Bob's valuation $b \in \mathcal{T}_B$. Certainly Alice will not want to deviate to an outcome $j \notin \{i, i+1\}$, as her utility for these outcomes is $-\infty$. Thus, to prove the "if" direction, it suffices to show that for each $i \in [n]$ and $b \in \mathcal{T}_B$, when transfers satisfy (*) for this value of $i$ and $b$, if Alice has a type $a_i \in \{a_{i,\ell}, a'_{i,\ell}, a_{i,h}, a'_{i,h}\}$, she will not want to deviate to the unique outcome in $\{i, i+1\} \setminus \{f(a_i, b)\}$. To prove the "only if" direction, it suffices to show that if transfers $p$ incentivize $f$, then (*) must hold for each $i \in [n]$ and $b \in \mathcal{T}_B$. To this end, consider any $i \in [n]$.

First, suppose $b_i = 0$. This means that $a_{i,\ell}$ receives $i$, and $a'_{i,\ell}$, $a_{i,h}, a'_{i,h}$ receive $i+1$.

Suppose that transfers $p$ satisfy (*), i.e. $p(i,b) - p(i+1,b) \in [-2^{-n},\ 2^{-n}]$. First, note that $a_{i,h}$ and $a'_{i,h}$ will not want to deviate to $i$, because these types have much higher utility for $i+1$ (and receive almost the same transfer on these two outcomes). Second, note that $u_A(a_{i,\ell}, i+1) - u_A(a_{i,\ell}, i) = -2^{-n}$ and $u_A(a'_{i,\ell}, i+1) - u_A(a'_{i,\ell}, i) = 2^{-n}$, and it follows by $p(i,b) - p(i+1,b) \in [-2^{-n},\ 2^{-n}]$ that

$$\begin{aligned} u_A(a_{i,\ell}, i) + p(i,b) &\geq u_A(a_{i,\ell}, i+1) + p(i+1,b)\\ u_A(a'_{i,\ell}, i+1) + p(i+1,b) &\geq u_A(a'_{i,\ell}, i) + p(i,b) \end{aligned}$$

Thus, $a_{i,\ell}$ and $a'_{i,\ell}$ will not want to deviate either.

Now we show that if transfers $p$ incentivize $f$, then they must satisfy (*) for this value of $i$. Observe that $a_{i,\ell}$ and $a'_{i,\ell}$ have almost the same utility for $i$ and $i+1$, yet receive different outcomes. This will force $p(i,b) - p(i+1,b) \in [-2^{-n},\ 2^{-n}]$. Specifically, for neither of $a_{i,\ell}$ nor $a'_{i,\ell}$ to want to deviate to each other, we must have

$$\begin{aligned} 2^{-n} + p(i,b) &= u_A(a_{i,\ell}, i) + p(i,b)\\ &\geq u_A(a_{i,\ell}, i+1) + p(i+1,b) = p(i+1,b)\\ 2^{-n} + p(i+1,b) &= u_A(a'_{i,\ell}, i+1) + p(i+1,b)\\ &\geq u_A(a'_{i,\ell}, i) + p(i,b) = p(i,b), \end{aligned}$$

and thus $p(i,b) - p(i+1,b) \in [-2^{-n},\ 2^{-n}]$.

Second, suppose $b_i = 1$. This means that $a_{i,\ell}, a'_{i,\ell}, a_{i,h}$ receive $i$, and $a'_{i,h}$ receives $i+1$. The logic in this case is analogous to the first case.

Suppose that transfers $p$ satisfy (*), i.e. $p(i,b) - p(i+1,b) \in [2^i - 2^{-n},\ 2^i + 2^{-n}]$. First, note that $a_{i,\ell}$ and $a'_{i,\ell}$ will not want to deviate to $i+1$, because these types have almost the same utility for $i$ and $i+1$ (and receive a much higher transfer on $i$). Second, note that $u_A(a_{i,h}, i+1) - u_A(a_{i,h}, i) = 2^i - 2^{-n}$ and $u_A(a'_{i,h}, i+1) - u_A(a'_{i,h}, i) = 2^i + 2^{-n}$, and it follows by $p(i,b) - p(i+1,b) \in [2^i - 2^{-n},\ 2^i + 2^{-n}]$

that

$$u_A(a_{i,h}, i) + p(i,b) \geq u_A(a_{i,h}, i+1) + p(i+1,b)$$
$$u_A(a'_{i,h}, i+1) + p(i+1,b) \geq u_A(a'_{i,h}, i) + p(i,b)$$

Thus, $a_{i,h}$ and $a'_{i,h}$ will not want to deviate either.

Now we show that if transfers $p$ incentivize $f$, then they must satisfy (*). Observe that $a_{i,h}$ and $a'_{i,h}$ have almost the same utilities for $i$ and $i+1$, yet receive different outcomes. This will force $p(i,b) - p(i+1,b) \in [2^i - 2^{-n},\ 2^i + 2^{-n}]$. Specifically, for neither of $a_{i,h}$ nor $a'_{i,h}$ to want to deviate to each other, we must have

$$\begin{aligned} 2^{-n} + p(i,b) &= u_A(a_{i,h}, i) + p(i,b) \\ &\geq u_A(a_{i,h}, i+1) + p(i+1,b) = 2^i + p(i+1,b) \\ 2^i + 2^{-n} + p(i+1,b) &= u_A(a'_{i,h}, i+1) + p(i+1,b) \\ &\geq u_A(a'_{i,h}, i) + p(i,b) = p(i,b), \end{aligned}$$

and thus $p(i,b) - p(i+1,b) \in [2^i - 2^{-n},\ 2^i + 2^{-n}]$.

□

We now define transfers $p^*$ such that

$$p^*(i,b) = -\sum_{j=1}^{i-1} b_j 2^j.$$

For each $b \in \mathcal{T}_B$ and $i \in [n]$, we have $p^*(i,b) - p^*(i+1,b) \in [b_i 2^i - 2^{-n},\ b_i 2^i + 2^{-n}]$, and thus by Lemma 3.2, these transfers incentivize $f$. Thus, let $\mathcal{M}$ denote the mechanism which has Alice announce her type (using $O(\log(n))$ bits), tells that type to Bob, and then has Bob decide the outcome $i$ (using $O(\log(n))$ bits) and the transfer $p^*(i,b)$ for Alice (using $O(n)$ bits). This mechanism EPIC implements $f$ with communication cost $O(n)$.

On the other hand, consider any mechanism $\mathcal{M}$ which EPIC implements $f$. Let $p$ denote the transfers $\mathcal{M}$ gives to Alice. By Lemma 3.2 and telescoping sum, the transfers must satisfy $p(1,b) - p(n+1,b) \in [\sum_{j=1}^n (b_j 2^j - 2^{-n}),\ \sum_{j=1}^n (b_j 2^j + 2^{-n})]$ for all $b \in \mathcal{T}_B$. Notice that for sufficiently large $n$, $n2^{-n}$ is tiny, and hence, the intervals $[\sum_{j=1}^n (b_j 2^j - 2^{-n}),\ \sum_{j=1}^n (b_j 2^j + 2^{-n})]$ corresponding to distinct $b$'s are disjoint. Since there are $2^n$ distinct $b$'s, there are also $2^n$ distinct values of $p(1,b) - p(n+1,b)$. Suppose for contradiction that $\mathcal{M}$ computes $p$ using $o(n)$ bits of communication. Then there also exists a protocol which can compute $p(1,b) - p(n+1,b)$ with $o(n)$ communication, which is impossible because there are $2^n$ such values. Therefore, any EPIC implementation of $f$ must have communication cost $\Omega(n)$. This completes the proof.

□

*Discussion.* In the proof above, we showed that computing the transfers requires large amount of communication because the transfers require a large number of bits to represent. For two players, this is necessary. That is, in a two player environment, if a social choice function $f$ can be incentivized with transfers that can be represented with $K$ bits, then there exists an EPIC implementation with communication cost $CC(f) + K$. This implementation first has Alice and Bob compute the social choice function using an optimal protocol, which requires $CC(f)$ bits, and then has each player specify the transfer for the other player (as we recalled in the proof of Theorem 3.1, the transfers to Alice are determined solely by the outcome and Bob's type and vice versa).

We note that it is possible to modify the environment by giving Bob nontrivial utilities such that $f$ is the unique social choice function which maximizes the welfare $u_A(a,i) + u_B(b,i)$. Specifically, for each Bob type $b \in \mathcal{T}_B$, we define Bob's utility as $u_B(b,i) = -\sum_{j=1}^{i-1} b_j 2^j$ for each outcome $i \in [n+1]$, which is equal to $p^*(i,b)$ in the proof. In this modified environment, $f$ always returns the unique outcome which maximizes welfare. Notice that $p^*(i,b)$ then becomes the VCG transfer (up to an additive constant that can depend Bob's type) for Alice. If we also let Alice output the VCG transfer (up to an additive constant that can depend on Alice's type) $p'(i,a) := u_A(a,i)$ for Bob after the outcome is decided, then $p'$ along with $f$ is EPIC for Bob. Together, $p^*, p'$ give an EPIC implementation of $f$.

Finally, in the above modified environment where $f$ is welfare-maximizing, note that despite Alice's valuation being succinctly representable, her utilities are "high precision". This is necessary, because by [18, Proposition 2], if all the valuations in the environment have low precision, every welfare-maximizing social choice function has an EPIC implementation with only slightly more communication for computing the transfers. Moreover, Bob's type requires many bits to represent. This is also necessary, because if both players have succinct types, they can simultaneously output their types, after which the mechanism computes the correct outcome and charges VCG transfers.

# 4 EXPONENTIAL SEPARATION OF $CC^{EPIC}(f)$ AND $CC^{DSIC}(f)$

In this section, we construct a social choice function $f$ such that $CC^{EPIC}(f) = O(n)$, yet $CC^{DSIC}(f) = \exp(n)$.

## 4.1 Building Up to Our Construction

We walk through a list of examples of environments and social choice rules, trying to build to an exponential separation of the communication required to EPIC implement and DSIC implement the rules. The first example is a classical illustration of the difference between ex-post and dominant strategy implementations for extensive form games.

*4.1.1 Attempt One.* Consider a second price auction with two bidders, Alice and Bob, and a single item, such that Alice's and Bob's value for the item are integers in $\{1, 2, \ldots, 10\}$. If the auction is implemented as a direct revelation mechanism, then it is DSIC. However, suppose we first ask Alice her value, then tell that value to Bob and ask him to respond with his own value. This mechanism is no longer DSIC. For example, one strategy of Bob is to always say his value is 1, except when Alice bids 8, in which case he will say his value is 9. When Bob plays this strategy and Alice's true value is 8, Alice gets more utility by lying and bidding 9 than by telling the truth.

We note that the above strategy for Bob is "crazy" in the sense that it does not maximize his own utility, but serves mostly to incentivize non-truthful bidding by Alice. Moreover, this crazy strategy for Bob was possible only because Bob knew Alice's value and decided his response as a function of this value. Observe that,

for such a crazy strategy to work, Bob does not have to know Alice's value exactly. Intuitively and informally, the following two conditions suffice:

(a) Bob learns information about Alice's type.
(b) Bob has two possible responses, one which gives Alice high utility, and one which give Alice low utility.

Our next idea is to construct an instance where any low communication mechanism must satisfy Item a and Item b above. We first focus on Item a and try to devise an instance where any low-communication mechanism requires Bob to know something about Alice's valuation. For this, we embed the well-known "Index" problem from communication complexity in a welfare-maximization context. Recall that, in the Index problem, there is a parameter $K > 0$ such that Alice has an index $k \in [K]$ and Bob has a vector $X = (x_i)_{i\in[K]} \in \{0,1\}^K$, and the goal is to output the $k^{\text{th}}$ location in the vector $X$, i.e. $x_k$.

Intuitively, the importance of the Index problem lies in the fact the only way to efficiently solve this problem is for Alice to reveal a lot of information about her input. Specifically, first observe that the protocol where Alice sends $k$ to Bob, and Bob then simply outputs $x_k$, uses communication $O(\log K)$. However, it turns out that any protocol that does not reveal a lot of information about Alice's input to Bob must have communication $\Omega(K)$ (this can be formalized, see [21, *etc.*], although we do not need to do so here).

*4.1.2 Attempt Two.* Consider an auction where there are two bidders and an even number $m$ of items for sale. The bidders, Alice and Bob, are multi-minded[8] with interests as follows: Alice is interested in exactly two sets, a set $S \subseteq [m]$ of size $m/2$ that she values at 4, and the set $\overline{S}$ that she values at 1. Bob's valuation is such that for every subset $T \subseteq [m]$ of size $m/2$, he is interested in exactly one of the sets $T$ and $\overline{T}$, which he values at 5 (and he values the other set at 0). The social choice function $f$ outputs the welfare-maximizing allocation of items between Alice and Bob. That is, Bob gets whichever of $S$ or $\overline{S}$ he values at 5, and Alice gets the complement (which she values at either 4 or 1). Observe that $f$ is incentive compatible without transfers.

The direct revelation mechanism $\mathcal{M}_1$ (where Alice and Bob simultaneously reveal their entire type) is DSIC. In this mechanism, Bob does not learn anything about Alice's type, that is, Item a in Section 4.1.1 does not hold. However, the fact that Bob communicates his entire type means that $\mathcal{M}_1$ requires communication exponential in $m$.

There is also a mechanism $\mathcal{M}_2$ for the above instance where Item a is satisfied. This is the mechanism that first asks Alice for the set $S$ of size $m/2$ she values at 4, and then asks Bob which of the sets $S$ and $\overline{S}$ he values at 5. The mechanism $\mathcal{M}_2$ then gives Bob the set he said he values at 5 and gives Alice the complement. Observe that $\mathcal{M}_2$ is EPIC and requires $O(m)$ communication. However, the mechanism $\mathcal{M}_2$ is not DSIC. Indeed, consider a (crazy) strategy for Bob where he always says that the set $S$ reported by Alice is the one he values at 5 (regardless of his input). With this strategy for Bob, Alice always gets the complement of what she reports and therefore, she is incentivized to lie and report the set $\overline{S}$ instead of the set $S$ which is truly her favorite.

*A low communication DSIC mechanism.* However, the instance above does not yield a separation between the communication complexity of DSIC and EPIC implementations, as there is an $O(m)$-communication mechanism that is also DSIC. This mechanism, which we we call $\mathcal{M}^\star$, asks Alice only report the sets $\{S, \overline{S}\}$ of size $m/2$ she has non-zero value for, without specifying which one of the two she values at 4. Then, the mechanism $\mathcal{M}^\star$ asks Bob which of the sets $S$ and $\overline{S}$ he values at 5, gives him that set and gives Alice the complement of the set.

The mechanism $\mathcal{M}^\star$ clearly has communication $O(m)$. It is DSIC, because if Alice reports anything other than than $\{S, \overline{S}\}$, she will get utility 0 regardless of what Bob says. In particular, it is not possible to construct a "crazy" strategy of Bob as in $\mathcal{M}_2$, because Bob's response cannot depend on the difference between $S$ and $\overline{S}$.

In other words, the reason the mechanism $\mathcal{M}^\star$ is DSIC is that it does not satisfy Item b above. Even though Bob learns a lot of information about Alice's type, he cannot respond to this information in a way that gives Alice a lower utility in some cases, and a higher utility in other cases.

*Need for new ideas.* It may seem at first that the mechanism $\mathcal{M}^\star$ works only because in our instance, Bob does not need to which of $S$ and $\overline{S}$ does Alice value at 4 in order to determine the welfare-maximizing allocation. However, this is not the case. Even if the welfare-maximizing allocation was dependent on which of $S$ and $\overline{S}$ is valued at 4 by Alice, Bob could just send two answers, one for the case when $S$ is valued at 4 and the other one for the when $\overline{S}$ is valued at 4. The resulting mechanism would still be DSIC. Thus, new ideas are needed to get a separation between the communication complexity of EPIC and DSIC implementations.

## 4.2 Construction and Intuition

At a high level, our main construction is simply two independent copies of the instance described in Section 4.1.2, where the valuation functions for Alice and Bob are additive over the two copies.

Formally, for every even $m$, we have a two player combinatorial auction where a set $M_1 \sqcup M_2$ of items satisfying $|M_1| = |M_2| = m$ is for sale. The set of outcomes is defined by[9]

$$Y = \{(X_1, X_2) \mid X_1 \subseteq M_1,\ X_2 \subseteq M_2,\ |X_1| = |X_2| = m/2\}.$$

An outcome $(X_1, X_2)$ indicates that Alice receives $(X_1, X_2)$ and Bob receives $(\overline{X_1}, \overline{X_2})$. Alice's types are also given by the set $\mathcal{T}_A = Y$ and her utility function $u_A : \mathcal{T}_A \times Y \to \mathbb{R}$ is defined by $u_A((S_1, S_2), (X_1, X_2)) = u_{A,1}(S_1, X_1) + u_{A,2}(S_2, X_2)$, where, for $i \in [2]$, we have:

$$u_{A,i}(S_i, X_i) = \begin{cases} 4, & \text{if } S_i = X_i \\ 1, & \text{if } S_i = \overline{X_i} \\ 0, & \text{otherwise.} \end{cases}$$

Bob's type set $\mathcal{T}_B$ is the collection of all pairs $(v_{B,1}, v_{B,2})$, where for $i \in [2]$, the function $v_{B,i}$ maps a subset of $M_i$ of size $m/2$ to the set

[8] Recall that a valuation function $v$ on $[m]$ is multi-minded if there exists a collection $\{(v_i, T_i)\}_i$, where each $v_i \in \mathbb{R}$ and $T_i \subseteq [m]$, such that $v(S) = \max\{v_i | T_i \subseteq S\}$. The sets $T_i$ are call the "interests" of the valuation function $v$.

[9] We restrict the auction to always award half of the items in $M_i$ to each bidder, for each $i \in [2]$. This restriction is without loss of generality, because the social choice function $f$ always outputs allocations with this property, but it simplifies the notation slightly.

$\{0, 5\}$ such that for each set $T \subseteq M_i$, $|T| = m/2$, we have $v_{B,i}(T) = 5$ and $v_{B,i}(\overline{T}) = 0$ or vice-versa. Bob's utility function is:

$$u_B((v_{B,1}, v_{B,2}), (X_1, X_2)) = v_{B,1}(\overline{X_1}) + v_{B,2}(\overline{X_2}).$$

Finally, the goal of the auctioneer is to maximize the welfare. Observe that, if Alice's type is $(S_1, S_2) \in \mathcal{T}_A$ and Bob's type is $(v_{B,1}, v_{B,2}) \in \mathcal{T}_B$, this corresponds to computing the outcome $(X_1, X_2)$, where, for $i \in [2]$, $X_i = S_i$ if $v_{B,i}(\overline{S_i}) = 5$ and $\overline{S_i}$ otherwise. For the rest of this section, let $f$ denote this social choice function.

*High Level Intuition.* We use the instance above to separate the communication complexity of EPIC and DSIC implementations. First, we consider the mechanism $\mathcal{M}^{\text{par}}$ which runs two instances of the mechanism $\mathcal{M}^\star$ from Section 4.1.2 in parallel. More formally, in the first round we ask Alice to report $\{S_1, \overline{S_1}\}$ and $\{S_2, \overline{S_2}\}$, without differentiating between sets up to complements. Bob then picks the allocation on both sets of items in round two. This mechanism EPIC implements $f$ with communication cost $O(m)$. However, as we show next, $\mathcal{M}^{\text{par}}$ fails to be DSIC.

Observe the following crucial detail of the social choice environment: when Alice's true type is $(S_1, S_2)$, Alice has utility 4 when she receives $(S_1, T_2)$ for any $T_2 \notin \{S_2, \overline{S_2}\}$, but she has utility 2 when she receives $(\overline{S_1}, \overline{S_2})$. This motivates us to construct the following strategy $s_B$ of Bob in $\mathcal{M}^{\text{par}}$: for some fixed sets $S_1^* \subseteq M_1, S_2^* \subseteq M_2$, if Alice reports $\{S_1^*, \overline{S_1^*}\}$ and $\{S_2^*, \overline{S_2^*}\}$ in round one, then Bob will give Alice $(\overline{S_1^*}, \overline{S_2^*})$. But if Alice reports $\{S_1^*, \overline{S_1^*}\}$ and $\{T_2, \overline{T_2}\}$ in round one, for any $T_2 \notin \{S_2^*, \overline{S_2^*}\}$, then Bob will give Alice $(S_1^*, T_2')$ (for $T_2' \in \{T_2, \overline{T_2}\}$ chosen arbitrarily). When Alice's true type is $(S_1^*, S_2^*)$, truth telling is not a best response of Alice against this strategy $s_B$. Thus, $\mathcal{M}^{\text{par}}$ is not DSIC.

We now argue informally that the existence of a "crazy" strategy like this for Bob is not an accident, but a property which is necessary in any communication efficient mechanism. Intuitively, this is because for the mechanism to be efficient, Alice must reveal a lot of information about both sets of items (implementing Item a from Section 4.1.1). Regardless of the order in which this is done, at the first point Bob learns about Alice's type on one set of items $M_i$, he can condition his response on the other set of items $M_{3-i}$ based on the information from $M_i$ This allows him to give Alice two sets she values at 1 when she tells the truth, yet at least one set which she values at 4 when she deviates (implementing Item b).

For a concrete example, we can also consider $\mathcal{M}^{\text{seq}}$, which denotes the mechanism which runs $\mathcal{M}^\star$ on the first set of items $M_1$, commits to the allocation on $M_1$, then runs $\mathcal{M}^\star$ on the second set of items $M_2$. Then the same argument as for $\mathcal{M}^{\text{par}}$ shows that there is a strategy of Bob against which truth telling is not a best response. However, we now need to change the argument so that Bob conditions his response on $M_2$ on Alice's actions on $M_1$, because Bob commits to a result on $M_1$ before he acts on $M_2$. Because Alice must reveal lots of information about her type on *both* $M_1$ and $M_2$, this argument should go through in any communication efficient mechanism.

## 4.3 Technical Considerations and Difficulties

In Section 4.2, we argued informally that at the earliest where Alice reveals information, it should be possible to construct a strategy of Bob against which truth telling is not a best response for Alice. Unfortunately, this is not literally true for every mechanism, and our proof must circumvent this fact. In this section, we first explain in more detail how such "crazy" strategies are constructed, and demonstrate that the needed "crazy" strategy cannot necessarily be constructed at the first node where Alice acts.

Consider a communication efficient mechanism $\mathcal{M} = (G, S_A, S_B)$, and for simplicity assume that $\mathcal{M}$ is perfect information[10]. This assumption allows us to not worry about situations where the mechanism asks Alice for information, but does not reveal all of that information to Bob.

Our goal is to construct a "crazy strategy" of Bob, against which truth-telling is not a best response for Alice. To construct this strategy, we want to find a node $h$ in the game tree of $\mathcal{M}$ where Alice communicates information which Bob can respond to in the following way: when Alice tells the truth, Bob must be able to give Alice a bad result, but if Alice deviates from truth telling, Bob can give Alice a good result on at least one of the sets of items. To explain this fully, we use the language of Section A.1. Specifically, we use $\mathcal{T}_A(h), \mathcal{T}_B(h)$ to denote the types of Alice and Bob for which the computation of $G$ under truth-telling passes through $h$. We need $h$ to satisfy the following:

(A) Alice acts at $h$, and there exist two of Alice's types $(S_1, S_2)$, $(T_1, T_2) \in \mathcal{T}_A(h)$ at $h$ such that $S_A((S_1, S_2))(h) \neq S_A((T_1, T_2))(h)$ (that is, $(S_1, S_2)$ and $(T_1, T_2)$ take different actions at $h$ under truth telling), and moreover, we either have $S_1 = T_1$ or $S_2 = T_2$. For concreteness, suppose that $S_1 = T_1$.

(B) There exist types $(v_{B,1}, v_{B,2}), (v'_{B,1}, v'_{B,2}) \in \mathcal{T}_B(h)$ such that $v_{B,1}(S_1) = 5, v_{B,1}(S_2) = 5$, and $v'_{B,1}(S_1) = 0$.

These correspond to Item a and Item b of Section 4.1.1, instantiated for the specific social choice function $f$.

Claim 4.1. *If there exists a node $h$ at which Item A and Item B are both satisfied, then $\mathcal{M}$ is not DSIC.*

Proof. Define a "crazy strategy" of Bob as follows: Bob acts according to $(v_B^1, v_B^2)$ in all nodes except those in the subtree where Alice plays the action chosen by $(T_1, T_2)$ at $h$, where Bob acts according to $(v_B^{1\prime}, v_B^{2\prime})$. Suppose Alice's true type is $(S_1, S_2)$. When Bob plays the above strategy and Alice tells the truth, Alice receives $(\overline{S_1}, \overline{S_2})$, which she values at 2. But if Alice deviates and plays strategy corresponding to $(T_1, T_2)$, then she receives $S_1$ on $M_1$, and receives a utility of 4. Thus, truth-telling is not a best response for Alice with type $(S_1, S_2)$, and $\mathcal{M}$ is not DSIC. □

Neither of the above conditions Item A or Item B on node $h$ are very strong independently. For example, at any node $h$ which is the first time Alice takes a nontrivial action, Item A will be satisfied for some set $(S_1, S_2)$. Furthermore, Item B will be satisfied at the root node of the game tree for *every* Alice type $(S_1, S_2)$. However, together these two requirements become somewhat subtle. Before we proceed to the formal proof, we highlight two cases of this subtlety, and briefly hint at how we address them.

[10] We prove in the appendix of the full version that this assumption is without loss of generality for our specific social choice function $f$.

(i) Suppose the first thing the mechanism does is ask Bob "is your type $(v^*_{B,1}, v^*_{B,2})$?" (for some $(v^*_{B,1}, v^*_{B,2})$ fixed by the mechanism). If the answer is yes, then all types of Alice have a dominant strategy in the corresponding subtree. Moreover, if the first question is to just ask Bob "is your full type on $M_1$ equal to $v^*_{B,1}$?" (for some fixed $v^*_{B,1}$, regardless of his type on $M_2$) then it is possible that Alice always has a dominant strategy in that subtree[11]. This shows that we cannot hope to construct the needed "crazy strategy" of Bob in every subtree of the game.

(ii) Suppose the first question is to ask Bob "what is your value on sets $\{T^*_1, \overline{T^*_1}\} \subseteq M_1$ and sets $\{T^*_2, \overline{T^*_2}\} \subseteq M_2$ (for some set $T^*_1, T^*_2$ fixed by the mechanism). At the (four nodes of the) next layer of the tree, ask Alice "Do you have $S_1 \in \{T^*_1, \overline{T^*_1}\}$ AND $S_2 \in \{T^*_2, \overline{T^*_2}\}$?" It turns out that truth-telling is a dominant action at every node in the first layer where Alice acts[12]. This shows that we cannot hope to construct the needed "crazy strategy" of Bob at every layer of the game tree.

Intuitively, we address the first issue by noting that, because Bob has more types than there are nodes in the game tree, we can safely ignore any node in which Bob has few types. We fix the second issue by changing the proof outline overall. Instead of taking an efficient mechanism $\mathcal{M}$ and finding a node $h$ satisfying Item A and Item B (thus showing that $\mathcal{M}$ is not DSIC), we use a proof by contradiction. Intuitively, we consider an efficient mechanism in which no such "crazy strategy" of Bob be constructed, and show that the questions such a mechanism can ask to Alice are so restrictive that the mechanism cannot possibly handle all types Alice might have.

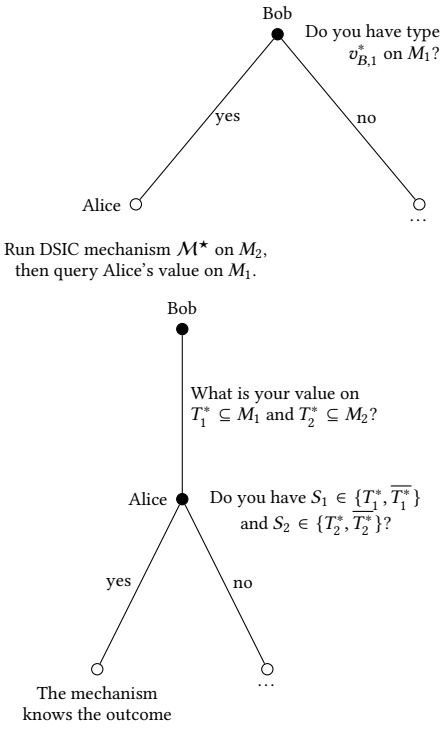


**Figure 1: Examples of the technical difficulties our proof needs to handle. These figures illustrate the first few layers of the game trees, while the remainder of the game is unspecified. Regardless of how the rest of the game computes $f$, these examples illustrate that the required strategy of Bob cannot always be constructed based on the first node where Alice acts (i.e. the first node where Alice acts may not satisfy Item A and Item B for any $(S_1, S_2), (T_1, T_2)$).**

## 4.4 Separation of DSIC and EPIC without Transfers

We now prove that, without transfers, social choice function $f$ from Section 4.2 requires an exponential amount of communication to implement in dominant strategies.

Theorem 4.2. *Any DSIC implementation of $f$ without transfers has communication cost $\widetilde{\Omega}(2^m)$.*

Proof. Fix a DSIC mechanism $\mathcal{M}$ and let $C_{\mathcal{M}}$ be the communication of $\mathcal{M}$. At the cost of blowing up the communication by a factor of two, we can assume by Lemma B.1 (in the full version of this paper) that $\mathcal{M}$ is perfect information. Let $\mathcal{M} = (G, S_A, S_B)$, i.e. $G$ is the perfect information extensive form game used by $\mathcal{M}$, and $S_A, S_B$ are the dominant type-strategy profile implementing $f$. By Section A.1, each node $h$ of the game tree $G$ corresponds to a set of types $\mathcal{T}_A(h)$ of Alice and $\mathcal{T}_B(h)$ of Bob, and each action taken at $h$ corresponds to partitioning the types of the player $\mathcal{P}(h) \in \{A, B\}$ who acts at this node. This partition is given by $S_i(\cdot)$, specifically, for each node $h'$ immediately after $h$ in $G$, a type $t \in \mathcal{T}_i(h)$ remains in $\mathcal{T}_i(h')$ if and only if $S_i(t)$ plays the action at $h$ which corresponds to $h'$. We say that type $t$ takes action $a$ at $h$ if $S_i(t)$ plays $a$ at $h$. Observe that for each $i$ and nodes $h, h'$ where $h$ is an ancestor of $h'$, we have $\mathcal{T}_i(h') \subseteq \mathcal{T}_i(h)$.

Define $K := \binom{m}{m/2}/2$ and observe that $K = \widetilde{\Theta}(2^m)$. Observe that, for $i \in [2]$, we can partition all subsets of $M_i$ of size $m/2$ into (unordered) pairs of the form $(T, \overline{T})$, and there will be exactly $K$ such pairs. Call these pairs $P_{i,1}, \cdots, P_{i,K}$ in some canonical order. For the rest of this section, we equivalently view a type $(v_{B,1}, v_{B,2})$ of Bob as a pair of bit-strings $B = (B_1, B_2) \in \{0,1\}^K \times \{0,1\}^K$, where $B_{i,k}$ for $k \in [K]$ specifies which set in $P_{i,k}$ Bob values at 5. Similarly, we can view Alice's type $(S_1, S_2)$ as a tuple $(k_1, k_2, b_1, b_2)$ where $k_1, k_2 \in [K]$ are indices and $b_1, b_2$ are bits, and, for $i \in [2]$, $S_i$ is the $b_i^{\text{th}}$ element in $P_{i,k_i}$. As $b_1, b_2$ are irrelevant to the outcome of the

[11] Observe that Alice already knows what will happen on $M_1$. Thus, in this subtree the mechanism can thus run the DSIC mechanism $\mathcal{M}^\star$ described in Section 4.1.1 on $M_2$. Then, as a final step the mechanism can ask Alice her type on $M_1$. Intuitively, Alice already knows what will happen on $M_1$ (and can always grantee her best attainable outcome on $M_1$ at the end), so she might as well try to get her full value on $M_2$.

[12] Formally, truth-telling is a dominant action at node $h$ if $S_A(t_A)$ gets utility at least as high as all strategies $s'_A$ such that $s'_A(h) \neq S_A(t_A)(h)$.

Clearly Alice has a dominant strategy if indeed she should answer "yes" in this layer. If not, either one or both of her sets are not in the specified pair. If both are not, she gets zero utility from lying. If one of her sets is in the specified pair, the outcome on the matching set of items is already fixed, so Alice might as well "continue" (answering "no") and hope for more utility on the other set of items, knowing she can always grantee her utility on the matching set of items.

mechanism, using Lemma B.3 (in the full version)[13], we can assume without loss of generality that for all $k_1, k_2 \in [K]$, and nodes $h$, we either have that all Alice's types of the form $(k_1, k_2, \cdot, \cdot) \in \mathcal{T}_A(h)$ or all types of the form $(k_1, k_2, \cdot, \cdot) \notin \mathcal{T}_A(h)$. Thus, when talking about the sets $\mathcal{T}_A(h)$ for nodes $h$, we can view Alice's type as simply a pair of indices $(k_1, k_2)$. We adopt this convention for the rest of this proof, and we consider $\mathcal{T}_A(h) \subseteq [K] \times [K]$. Correspondingly, we consider $S_A(\cdot)$ to be a map from $[K] \times [K]$ to strategies in $\mathcal{M}$, and refer to the actions taken by pairs $(k_1, k_2) \in [K] \times [K]$.

The social choice function $f$ is determined by Alice's index on both sets of items, as well as Bob's value on those two indices. Thus, for each leaf $\ell$, we have $\mathcal{T}_A(\ell) = \{(k_1, k_2)\}$ a singleton[14], which will be a key observation in our proof.

We now begin to build the language and tools needed to address the considerations highlighted in Section 4.3.

Definition 4.3 (Shattered pairs). *Let $h$ be a node and $(k_1, k_2) \in [K] \times [K]$. We say that $(k_1, k_2)$ is* shattered *at $h$ if Bob's types $B = (B_1, B_2) \in \mathcal{T}_B(h)$, when restricted to coordinates $k_1, k_2$, take on all four possible values. In other words,*

$$\left|\{(B_{1,k_1}, B_{2,k_2}) \mid (B_1, B_2) \in \mathcal{T}_B(h)\}\right| = 4.$$

*We use $\mathcal{T}^{\text{Sh}}(h) \subseteq [K] \times [K]$ to denote the set of all pairs $(k_1, k_2)$ that are shattered at $h$.*

For convenience, we define the "neighbors" of a pair $(k_1, k_2)$ to be all those pairs with (at least) one index in common with $(k_1, k_2)$. Note that $(k_1, k_2) \in \text{nbr}(k_1, k_2)$.

Definition 4.4 (Neighbors). *Let $(k_1, k_2) \in [K] \times [K]$. A* neighbor *of $(k_1, k_2)$ is any pair of the form form $(k_1, k')$ or $(k', k_2)$ for some $k' \in [K]$. We use $\text{nbr}(k_1, k_2)$ to denote the set of all neighbors of $(k_1, k_2)$.*

Our first two lemmas show that a pair being shattered at a node $h$ severely restricts which questions the mechanism can ask at $h$. The first lemma corresponds to Claim 4.1, recast in the language of this proof. More specifically, Item B from Section 4.3 corresponds to a pair $(k_1, k_2)$ being shattered at $h$, and Item A from Section 4.3 corresponds to $(k_1, k_2)$ taking a different action than one of its neighbors. These two items cannot simultaniously occur in a DSIC mechansism.

Lemma 4.5. *Consider any node $h$ with $\mathcal{P}(h) = A$ and shattered pair $(k_1, k_2) \in \mathcal{T}^{\text{Sh}}(h) \cap \mathcal{T}_A(h)$. Then every pair in $\text{nbr}(k_1, k_2) \cap \mathcal{T}_A(h)$ must take the same action as $(k_1, k_2)$ at $h$.*

Proof. Fix a $(k_1, k_2) \in \mathcal{T}^{\text{Sh}}(h) \cap \mathcal{T}_A(h)$. Suppose for contradiction that there exists a neighbor of $(k_1, k_2)$ which is in $\mathcal{T}_A(h)$, yet takes a different action from $(k_1, k_2)$ at $h$. Without loss of generality, assume this neighbor is of the form $(k_1, k_2')$. We derive a contradiction by constructing a "crazy" strategy for Bob, exactly as in Claim 4.1, that violates the DSIC property.

Pick some $(B_1, B_2), (B_1', B_2') \in \mathcal{T}_B(h)$ with $B_{1,k_1} = B_{2,k_2} = 1$ and $B'_{1,k_1} = 0$ (these exist by Definition 4.3). We define a strategy $s_B$ of Bob such that in subtree where Alice plays the action taken by $(k_1, k_2')$ at $h$, $s_B$ plays the action played by $S_B((B_1', B_2'))$. In every other node of the game tree, $s_B$ plays the same action played by $S_B((B_1, B_2))$. This completely specifies $s_B$.

Suppose Alice has type $(k_1, k_2, 1, 1)$ (that is, for $i \in [2]$, her desired sets are in $P_{i,k_i}$, and her most preferred set is the one from $P_{i,k_i}$ which $(B_1, B_2)$ values at 5). When Alice plays $S_A((k_1, k_2))$, she is allocated her less preferred set on both $M_1$ and $M_2$, and thus gets utility 2. But if Alice deviates and plays $S_A((k_1, k_2'))$, on $M_1$ she receives her most preferred set, which she values at 4. Thus, truth telling is not a best response for Alice with type $(k_1, k_2)$, and thus $\mathcal{M}$ is not DSIC. □

We just showed that if some pair is shattered and lies in some $\mathcal{T}_A(h)$, then all of its neighbors who are also in $\mathcal{T}_A(h)$ must take the same action. Next, we need something stronger, namely that all shattered pairs at $h$ take the same action. Along the way we prove that additionally that if a pair is shattered and lies in $\mathcal{T}_A(h)$, then all of its neighbors must lie in $\mathcal{T}_A(h)$.

Lemma 4.6. *Let $h$ be any node, and consider the set*

$$\mathcal{T}_A^{\text{Sh-nbr}}(h) := \bigcup_{(k_1,k_2) \in \mathcal{T}^{\text{Sh}}(h) \cap \mathcal{T}_A(h)} \text{nbr}(k_1, k_2).$$

*Then we have that $\mathcal{T}_A^{\text{Sh-nbr}}(h) \subseteq \mathcal{T}_A(h)$. Moreover, if $\mathcal{P}(h) = A$, then all types in $\mathcal{T}_A^{\text{Sh-nbr}}(h)$ must take the same action at $h$.*

Proof. First, we prove that $\mathcal{T}_A^{\text{Sh-nbr}}(h) \subseteq \mathcal{T}_A(h)$. Suppose for contradiction that this is not the case. This means that there exists $(k_1, k_2) \in \mathcal{T}^{\text{Sh}}(h) \cap \mathcal{T}_A(h)$ and $(k_1', k_2') \in \text{nbr}(k_1, k_2)$, but $(k_1', k_2') \notin \mathcal{T}_A(h)$.

Consider the node $h'$ which is the latest Alice node along the path from the root to $h$ at which $(k_1', k_2') \in \mathcal{T}_A(h')$. By definition, $(k_1, k_2)$ and $(k_1', k_2')$ take different actions at $h'$. Observe that, because $\mathcal{T}_B(h) \subseteq \mathcal{T}_B(h')$, we also have $\mathcal{T}^{\text{Sh}}(h) \subseteq \mathcal{T}^{\text{Sh}}(h')$, and thus $(k_1, k_2)$ is shattered at $h'$. But then, by Lemma 4.5, $(k_1, k_2)$ and $(k_1', k_2')$ must take the same action at $h'$, a contradiction.

Conclude using $\mathcal{T}_A^{\text{Sh-nbr}}(h) \subseteq \mathcal{T}_A(h)$ and Lemma 4.5 that, if $(k_1, k_2) \in \mathcal{T}^{\text{Sh}}(h) \cap \mathcal{T}_A(h)$, then each pair in $\text{nbr}(k_1, k_2)$ takes the same action at $h$. Thus, to prove that all types in $\mathcal{T}_A^{\text{Sh-nbr}}(h)$ take the same action, it suffices to show that if $(k_1, k_2), (k_1', k_2') \in \mathcal{T}^{\text{Sh}}(h) \cap \mathcal{T}_A(h)$ with $k_1 \neq k_1'$ and $k_2 \neq k_2'$, then $(k_1, k_2)$ and $(k_1', k_2')$ must still take the same action at $h$.

To prove this, we make use of the fact that $\mathcal{T}_A^{\text{Sh-nbr}} \subseteq \mathcal{T}_A(h)$. In particular, means that $(k_1, k_2') \in \text{nbr}(k_1, k_2) \subseteq \mathcal{T}_A(h)$. By Lemma 4.5, $(k_1, k_2')$ must take the same action as $(k_1, k_2)$. By the exact same logic, $(k_1, k_2')$ must take the same action as $(k_1', k_2')$. Thus, $(k_1, k_2)$ and $(k_1', k_2')$ take the same action at $h$, and so does every pair in $\mathcal{T}_A^{\text{Sh-nbr}}(h)$.

□

While $\mathcal{T}^{\text{Sh}}(h) \subseteq [K] \times [K]$ is defined entirely in terms of Bob's types, $\mathcal{T}_A^{\text{Sh-nbr}}(h) \subseteq \mathcal{T}_A(h)$ tells us information about Alice's types as well. This provides us with a convenient way to describe the rest of the proof, in terms of the following observation:

[13] More formally, consider the partition of Alice's types given by $\{\{(k_1, k_2, b_1, b_2)\}_{(b_1,b_2)\in\{0,1\}\times\{0,1\}}\}_{(k_1,k_2)\in[K]\times[K]}$. For all fixed types of Bob, $f$ is constant on the above partition, and thus by Lemma B.3 in the full version, we can assume that for all $h$, if one element of a set $\{(k_1, k_2, b_1, b_2)\}_{(b_1,b_2)\in\{0,1\}\times\{0,1\}}$ is in $\mathcal{T}_A(h)$, then all elements of that set are in $\mathcal{T}_A(h)$.

[14] Bob's type, on the other hand, need only be determined on indices $k_1, k_2$. That is, for each leaf $\ell$, if we have $\mathcal{T}_A'(\ell) = \{(k_1, k_2)\}$, then for each $(B_1, B_2), (B_1', B_2') \in \mathcal{T}_B(\ell)$, we have $B_{1,k_1} = B'_{1,k_1}$ and $B_{2,k_2} = B'_{2,k_2}$.

Observation 4.7. *For all leaves $\ell$ of the game tree, we have $\mathcal{T}_A^{\text{Sh-nbr}}(\ell) = \emptyset$ (that is, $\mathcal{T}^{\text{Sh}}(\ell) \cap \mathcal{T}_A(\ell) = \emptyset$).*

Proof. Recall that social choice function $f$ is determined by Alice's index on both sets of items, as well as Bob's value on those two indices. In particular, at every leaf node, the mechanism must completely know Alice's pair in order to correctly compute $f$. Thus, for all leaves $\ell$ of the game tree, $\mathcal{T}_A(\ell)$ is a singleton. Using Lemma 4.6 and the fact that $|\text{nbr}(k_1, k_2)| > 1$, this is possible only if all leaves $\ell$ satisfy $\mathcal{T}_A^{\text{Sh-nbr}}(\ell) = \emptyset$. □

Our task in the remainder of the proof is to show that, if the communication cost $C_{\mathcal{M}}$ of $\mathcal{M}$ is sufficiently small, then there must exist a leaf $\ell$ with $\mathcal{T}_A^{\text{Sh-nbr}}(\ell) \neq \emptyset$.

Note that it is possible for a mechanism with exponential communication to satisfy $\mathcal{T}_A^{\text{Sh-nbr}}(\ell) = \emptyset$ at every leaf. Indeed, in the direct revelation mechanism where Bob reveals his entire type, $\mathcal{T}_B(\ell)$ is singleton for every leaf, and thus $\mathcal{T}^{\text{Sh}}(\ell) = \emptyset$. However, our next two lemmas shows that in low-communication mechanisms, very few types of Bob can ever end up at leafs $\ell$ in which $|\mathcal{T}^{\text{Sh}}(\ell)|$ is small. This serves to address Item i from Section 4.3. Our next lemma is a standard communication complexity argument, and intuitively states that "typical" Bob types always end up in leaves with a large number of Bob types.

Lemma 4.8. *Let $\mathcal{T}_B^{\text{Typ}} \subseteq \mathcal{T}_B$ denote the set of all Bob types $B \in \mathcal{T}_B$ such that for all leaves $\ell$ with $B \in \mathcal{T}_B(\ell)$, we have $|\mathcal{T}_B(\ell)| \geq |\mathcal{T}_B| \cdot 4^{-2C_{\mathcal{M}}}$. We have*

$$\left|\mathcal{T}_B^{\text{Typ}}\right| \geq |\mathcal{T}_B| \cdot (1 - 4^{-C_{\mathcal{M}}})$$

Proof. We show that $\left|\mathcal{T}_B \setminus \mathcal{T}_B^{\text{Typ}}\right| \leq 4^{-C_{\mathcal{M}}} \cdot |\mathcal{T}_B|$. Indeed, for all $B \in \mathcal{T}_B \setminus \mathcal{T}_B^{\text{Typ}}$, there exists a leaf $\ell$ such that $B \in \mathcal{T}_B(\ell)$ and $|\mathcal{T}_B(\ell)| < |\mathcal{T}_B| \cdot 4^{-2C_{\mathcal{M}}}$. There are at most $2^{C_{\mathcal{M}}}$ leaves in $G$. This gives:

$$\begin{aligned}\left|\mathcal{T}_B \setminus \mathcal{T}_B^{\text{Typ}}\right| &\leq \sum_{\substack{\text{leaf } \ell \text{ such that} \\ |\mathcal{T}_B(\ell)| < |\mathcal{T}_B| \cdot 4^{-2C_{\mathcal{M}}}}} |\mathcal{T}_B(\ell)| \\ &\leq \sum_{\text{leaf } \ell} |\mathcal{T}_B| \cdot 4^{-2C_{\mathcal{M}}} \leq |\mathcal{T}_B| \cdot 4^{-C_{\mathcal{M}}}.\end{aligned}$$

□

Next, we show that in all "typical nodes" (that is, nodes containing even one type from $\mathcal{T}_B^{\text{Typ}}$), *most* of the pairs in $[K] \times [K]$ are shattered. This follows from a combinatorial argument – if a lot of pairs are *not* shattered at $h$, then there cannot possibly be enough types in $\mathcal{T}_B(h)$ for $h$ to be "typical".

Lemma 4.9. *For any node $h$ such that $\mathcal{T}_B(h) \cap \mathcal{T}_B^{\text{Typ}} \neq \emptyset$, we have $\left|\mathcal{T}^{\text{Sh}}(h)\right| \geq K^2 - 10KC_{\mathcal{M}}$.*

Proof. Fix a node $h$ with $\mathcal{T}_B(h) \cap \mathcal{T}_B^{\text{Typ}} \neq \emptyset$. Some descendent of $h$ is a leaf node $\ell$ where $\mathcal{T}_B(\ell) \cap \mathcal{T}_B^{\text{Typ}} \neq \emptyset$. By the definition of $\mathcal{T}_B^{\text{Typ}}$, this means $|\mathcal{T}_B(\ell)| \geq |\mathcal{T}_B| \cdot 4^{-2C_{\mathcal{M}}}$. Thus $|\mathcal{T}_B(h)| \geq |\mathcal{T}_B| \cdot 4^{-2C_{\mathcal{M}}}$ as well. This condition will suffice to bound $\left|\mathcal{T}^{\text{Sh}}(h)\right|$.

We consider the set $\overline{\mathcal{T}^{\text{Sh}}(h)} = [K] \times [K] \setminus \mathcal{T}^{\text{Sh}}(h)$ of *unshattered* pairs, and proceed by showing that $|\overline{\mathcal{T}^{\text{Sh}}(h)}| \leq 10KC_{\mathcal{M}}$. To this end, observe that each unshattered pair $(k_1, k_2)$ can be uniquely written as $(k, k+d)$ for some values of $k, d \in [K]$ (where we take indexes mod $K$). To show that $|\overline{\mathcal{T}^{\text{Sh}}(h)}| \leq 10KC_{\mathcal{M}}$, we actually show that for all $d \in [K]$, the number of pairs of the form $(k, k+d) \in \overline{\mathcal{T}^{\text{Sh}}(h)}$ is at most $10C_{\mathcal{M}}$. Summing over all $d \in [K]$ then proves the lemma.

Fix a $d$ and suppose for contradiction that there were more than $R = 10C_{\mathcal{M}}$ pairs in $\overline{\mathcal{T}^{\text{Sh}}(h)}$ of the form $(k, k+d)$. For any fixed $d$, all of the $4^K$ types of Bob in $\mathcal{T}_B$ can be uniquely described by specifying Bob's value on $(k, k+d)$ for each $k \in [K]$, that is, by specifying for each $k \in [K]$ one of the four possible values of $(B_{1,k}, B_{2,k+d}) \in \{0,1\} \times \{0,1\}$.

If $(k, k+d)$ is shattered at $h$, then the types of Bob in $\mathcal{T}_B(h)$ can take on all 4 possible values on indexes $(k, k+d)$. However, if $(k, k+d)$ is unshattered at $h$, then there is at least one of the 4 options for Bob's type on indexes $(k, k+d)$ which never occurs in $\mathcal{T}_B(h)$. Thus, the types of Bob in $\mathcal{T}_B(h)$ can take on at most 3 possible values on indices $(k, k+d)$. Thus, the number of types in $\mathcal{T}_B(h)$ satisfies

$$|\mathcal{T}_B(h)| \leq 3^R \cdot 4^{K-R} = 4^{K-(1-\log_4 3)R} < 4^{K-(1/5)R}.$$

Plugging $R = 10C_{\mathcal{M}}$, we have $|\mathcal{T}_B(h)| < 4^{K-2C_{\mathcal{M}}} = |\mathcal{T}_B| \cdot 4^{-2C_{\mathcal{M}}}$, which contradicts what we know about $|\mathcal{T}_B(h)|$.

□

Even in communication efficient mechanisms, there can be leaf nodes with $\mathcal{T}_A^{\text{Sh-nbr}}(\ell) = \emptyset$. The right hand side of Figure 1, illustrating Item ii in Section 4.3, gives an example. Looking into this example deeper, we see the reason: at the node $h$ where Alice acts, if we take the action *not taken by the pairs in* $\mathcal{T}^{\text{Sh}}(h)$, then we arrive at a leaf node with $\mathcal{T}^{\text{Sh}}(\ell) \cap \mathcal{T}_A(\ell) = \emptyset$. Thus, intuitively, our approach for the remainder of this proof is to follow the actions taken by $\mathcal{T}^{\text{Sh}}(h)$ in order to arrive at a node with $\mathcal{T}_A^{\text{Sh-nbr}}(\ell) \neq \emptyset$.

We now begin to wrap up our proof. Assume for contradiction that $C_{\mathcal{M}} < K/10$. By Lemma 4.8, we get that $C_{\mathcal{M}} < K/10$ implies that $\mathcal{T}_B^{\text{Typ}} \neq \emptyset$. Fix an arbitrary $B^\star \in \mathcal{T}_B^{\text{Typ}}$. From Lemma 4.9, for any node $h$ such that $B^\star \in \mathcal{T}_B(h)$, we have $\mathcal{T}^{\text{Sh}}(h) \neq \emptyset$.

Define a collection of nodes $H^\star$ in $G$ as follows:

$$H^\star = \{h \mid B^\star \in \mathcal{T}_B(h),\ \mathcal{T}^{\text{Sh}}(h) \subseteq \mathcal{T}_A(h)\}$$

Observe that the root $h_0$ of $G$ is in $H^\star$ (because $\mathcal{T}_A(h_0) = \mathcal{T}_A$ and $\mathcal{T}_B(h_0) = \mathcal{T}_B$), and thus $H^\star \neq \emptyset$. Now, define $h^\star$ to any node in $H^\star$ for which no descendent of $h^\star$ is in $H^\star$[15].

First, we claim that $h^\star$ cannot be a node where Bob acts. Otherwise, consider the child $h'$ of $h^\star$ corresponding to the action taken by $B^\star$ at $h^\star$. At $h'$, Alice's type set remains the same, while sets $\mathcal{T}_B(h')$ and thus $\mathcal{T}^{\text{Sh}}(h')$ have only decreased from $h^\star$. Thus,

$$\mathcal{T}^{\text{Sh}}(h') \subseteq \mathcal{T}^{\text{Sh}}(h^\star) \subseteq \mathcal{T}_A(h^\star) = \mathcal{T}_A(h'),$$

and $h' \in H^\star$. This contradictions the choice of $h^\star$.

Next, we claim $h^\star$ cannot be a node where Alice acts. Suppose otherwise. Because $\mathcal{T}^{\text{Sh}}(h^\star) \subseteq \mathcal{T}_A(h^\star)$, we have $\mathcal{T}^{\text{Sh}}(h^\star) \subseteq \mathcal{T}_A^{\text{Sh-nbr}}(h^\star)$. Because $\mathcal{T}^{\text{Sh}}(h^\star) \neq \emptyset$, we have $\mathcal{T}_A^{\text{Sh-nbr}}(h^\star) \neq \emptyset$. By Lemma 4.6, there is thus a single child $h'$ of $h^\star$ such that every pair

[15] One can use Lemma 4.6 to show that $h^\star$ is unique and $H^*$ forms a path from the root to a leaf. However, this is not needed for our argument to go through.

in $\mathcal{T}_A^{\text{Sh-nbr}}(h^\star)$ takes the action leading to $h'$. As Bob's type set is unchanged at $h'$, we thus get

$$\mathcal{T}^{\text{Sh}}(h') = \mathcal{T}^{\text{Sh}}(h^\star) \subseteq \mathcal{T}_A^{\text{Sh-nbr}}(h^\star) \subseteq \mathcal{T}_A(h'),$$

and $h' \in H^\star$. This contradictions the choice of $h^\star$.

This means that $h^\star$ must be a leaf node. But then we have $\emptyset \neq \mathcal{T}^{\text{Sh}}(h^\star) \subseteq \mathcal{T}_A(h^\star)$, and thus $\mathcal{T}_A^{\text{Sh-nbr}}(h^*) \neq \emptyset$. This contradicts Observation 4.7.

Thus, for any mechanism $M$ which DSIC implements $f$ without transfers, $H_M \geq (1/10)K = \widetilde{\Omega}(2^m)$.

□

We remark that this theorem is tight up to constants, as the direct revelation mechanism asking Bob to reveal his entire type has communication cost $O(K)$.

## 4.5 Extension to the Case with Transfers

We have shown that social choice function $f$, which is incentive compatible without transfers, cannot be efficiently DSIC implemented without transfers. However, this does not yet rule out the existence of certain transfer functions which can efficiently DSIC implement $f$[16].

*Modified construction.* We now describe how to modify our construction to prove our separation even in the regime with transfers. For each $i \in [2]$ and each pair of subsets of items of the form $\{T, \overline{T}\} \subseteq M_i$ where $|T| = m/2$, we add a type of Alice which values every set at 0. The outcome when Alice has such a type on $M_i$ is identical to if Alice had positive utility for $T$ and $\overline{T}$ originally. That is, Bob receives whichever set among $\{T, \overline{T}\}$ he values at 5, and Alice receives the complement.

More formally, the set of outcomes and Bob's types (and Bob's utility for each outcome) remains unchanged, but Alice's type set changes. For $i \in [2]$, we let $\mathcal{Q}_i$ denote the collection of all subsets of $M_i$ of size $m/2$, and let $\mathcal{R}_i$ denote the collection of all unordered pairs of subsets of $M_i$ of the form $\{T, \overline{T}\}$ with $|T| = m/2$. Alice's new set of types are then $\mathcal{T}_A' = (\mathcal{Q}_1 \cup \mathcal{R}_1) \times (\mathcal{Q}_2 \cup \mathcal{R}_2)$. We use $\mathcal{R}_i$ to represent the cases where Alice gets 0 value from sets of items in $M_i$. Specifically, Alice's utility function $u_A' : \mathcal{T}_A' \times Y \to \mathbb{R}$ is now defined by $u_A'((S_1, S_2), (X_1, X_2)) = u_{A,1}'(S_1, X_1) + u_{A,2}'(S_2, X_2)$, where, for $i \in [2]$, we have:

$$u_{A,i}'(S_i, X_i) = \begin{cases} 4, & \text{if } S_i \in \mathcal{Q}_i \text{ and } S_i = X_i \\ 1, & \text{if } S_i \in \mathcal{Q}_i \text{ and } S_i = \overline{X_i} \\ 0, & \text{otherwise.} \end{cases}$$

In particular, $u_{A,i}'(S_i, X_i) = 0$ whenever $S_i \in \mathcal{R}_i$.

We define the social choice function $f'$ as follows: Let $f'\big((S_1, S_2), (v_{B,1}, v_{B,2})\big) = (X_1, X_2)$, where for $i \in [2]$, if $S_i \in \mathcal{R}_i$, then $X_i \in S_i$ is such that $v_{B,i}(\overline{X_i}) = 5$, and if $S_i \in \mathcal{Q}_i$, then $X_i = S_i$ if $v_{B,i}(\overline{S_i}) = 5$ and $X_i = \overline{S_i}$ if $v_{B,i}(S_i) = 5$. Observe that $f'$ is still welfare maximizing (although we use a very specific tie-breaking rule for those Alice types with $S_i \in \mathcal{R}_i$).

Intuitively, the addition of Alice types which are irrelevant to all outcomes allows us to say that the mechanism cannot award transfer to Alice in a nontrivial way. This allows us to reduce to the case without transfers. We make this formal below.

Theorem 4.10. *There exists an EPIC implementation of $f'$ with communication cost $O(m)$. However, any DSIC implementation of $f'$ with transfers has communication cost $\widetilde{\Omega}(2^m)$. That is,*

$$CC^{EPIC}(f') = O(m) \qquad CC^{DSIC}(f') = \widetilde{\Omega}(2^m).$$

Proof. Consider the mechanism which asks Alice to reveal her entire type, tells that type to Bob, and asks Bob to choose an outcome. All transfers are 0. This is EPIC, for the exact same reason that $\mathcal{M}^{\text{par}}$ in Section 4.2 is EPIC. Moreover, the communication cost is $O(m)$, as desired.

Now, consider a mechanism $\mathcal{M}_0$ which DSIC implements $f'$ with transfers.

In principle, this mechanism may provide nonzero transfers to Bob (specifically, if Alice acts at the root node, this action may change the transfer to Bob arbitrarily). However, observe that if we replace every transfer to Bob with 0, the result is still DSIC. This is because incentives have changed only for Bob, but Bob can now guarantee himself utility 10 when he follows $S_B(t_B)$ (and this is the highest utility he can achieve). Let $\mathcal{M}_1$ denote the mechanism that sets every transfer to Bob in $\mathcal{M}_0$ to 0.

Now that Bob has constant utility in every outcome selected by $\mathcal{M}_1$, let $\mathcal{M}_2$ denote the result of applying Lemma B.1 (from the full version of this paper) to $\mathcal{M}_1$ to get a perfect information DSIC mechanism. This leaves the transfers unchanged, and effects the communication cost by only a constant factor.

We describe and partition Alice's types $\mathcal{T}_A'$ in a similar way to how we partitioned them in the proof of Theorem 4.2. Let $K := \binom{m}{m/2}/2$ and, for $i \in [2]$, recall that $\mathcal{R}_i$ denotes a partition of subsets of $M_i$ of size $m/2$ into pairs $\{T, \overline{T}\}$. Index the pairs in this partition by $k \in [K]$. Then the type of Alice can be described by tuple $(k_1, k_2, v_1, v_2)$ where $k_1, k_2 \in [K]$ and $v_1, v_2 \in \{0, 1, 2\}$. Specifically, for $i \in [2]$, the index $k_i$ specifies which set in $\mathcal{R}_i$ Alice's type corresponds to, and $v_i \in \{0, 1, 2\}$ specifies whether Alice's most preferred set is $T$ (when $v_i = 0$), $\overline{T}$ (when $v_i = 1$), or neither (i.e. if $v_i = 2$, then Alice receives 0 utility from all subsets of $M_i$).

Observe that for every type of Bob, $f'$ is independent of the values $v_1, v_2$, and depends only on the indices $(k_1, k_2)$. That is, $f'$ is constant on every element of the partition of Alice's types given by

$$\{\{(k_1, k_2, v_1, v_2)\}_{v_1, v_2 \in \{0,1,2\}}\}_{(k_1, k_2) \in [K] \times [K]}.$$

We can thus apply Lemma B.3 (in the full version) to $\mathcal{M}_2$ to get a mechanism $\mathcal{M}_3$, which is still DSIC and has the same communication cost as $\mathcal{M}_2$, and additionally never distinguishes between types of the form $(k_1, k_2, \cdot, \cdot)$. Formally, in $\mathcal{M}_3$, if we have $(k_1, k_2, v_1, v_2) \in \mathcal{T}_A(h)$ for some $v_1, v_2$, then we have $(k_1, k_2, v_1, v_2) \in \mathcal{T}_A(h)$ for every $v_1, v_2$.

In principle, $\mathcal{M}_3$ can provides non-constant transfers to Alice. Specifically, Bob can act at the root node in a way which changes the transfer to Alice arbitrarily. However, it turns out that this

[16] Note that in principle it is possible for certain transfer functions to make a mechanism DSIC, but for others to render a mechanism EPIC but not DSIC. For example, suppose Alice has two types $L, R$ and Bob has two types $a, b$, and we have $f(L, a) = 1, f(L, b) = 2, f(R, a) = 3, f(R, b) = 4$. Let Alice with type $L$ value 1, 2, 3, 4 at 10, 7, 8, 1 respectively, and Alice with type $R$ value 1, 2, 3, 4 at 0, 0, 10, 10 respectively. Bob's valuations are irrelevant. Consider the perfect information mechanism sequentially asking Alice for her type, then Bob for his. If no transfers are included (that is, all transfers are 0) then this mechanism is EPIC but not DSIC. If the transfers to Alice when the outcome is 1, 2, 3, 4 are 0, 2, 0, 0 respectively (that is, we pay Alice 2 when outcome 2 is selected), then this mechanism is DSIC.

is all that is possible. Namely, there cannot be any Alice node $h$ with two leaf nodes, $\ell_1, \ell_2$ which are descendants of $h$, such that Alice receives different transfers in $\ell_1$ and $\ell_2$, If there were, Alice would have a strategic manipulation when her type is indifferent on both sets of items, i.e. when $v_1 = v_2 = 2$. Specifically, suppose the transfer at $\ell_1$ is higher than the transfer at $\ell_2$. Because the game is perfect information, there is some strategy of Bob such that, when Alice follows actions directed towards $\ell_1$, Bob takes actions directed towards $\ell_1$, and when Alice takes actions directed toward $\ell_2$, Bob takes actions directed towards $\ell_2$. Then, whichever Alice type takes $\ell_2$ has a strategic manipulation against this strategy of Bob when her type has $v_1 = v_2 = 2$.

Now, consider replacing every transfer to Alice in $\mathcal{M}_3$ with 0 to get a mechanism $\mathcal{M}_4$. By the reasoning in the preceding paragraph, the strategic situation for Alice has not changed at any node. More specifically, for each Alice node $h$, the transfers in $\mathcal{M}_3$ were constant at every leaf below $h$. This is still true in $\mathcal{M}_4$. Thus, if there were no strategic manipulations in $\mathcal{M}_3$, there can be no strategic manipulations in $\mathcal{M}_4$.

Thus, $\mathcal{M}_4$ constitutes an implementation of $f'$ in which the transfers to both agents are always 0. By simply ignoring the values of $v_1, v_2$, this constitutes an implementation of $f$ from Section 4.2. By Theorem 4.2, this means that the communication cost of $\mathcal{M}_4$ (and thus $\mathcal{M}_0$) is $\widetilde{\Omega}(2^m)$.

□

# A FORMAL DEFINITIONS AND PRELIMINARY ANALYSIS

*Environments and Implementations.* An *environment* for a set of $n$ *players* $i = 1, \dots, n$ is a tuple $E = (Y, \mathcal{T}_1, \dots, \mathcal{T}_n, u_1, \dots, u_n)$. Here, $Y$ is the set of *outcomes* and each $\mathcal{T}_i$ is the set of *types* of player $i$. Each $u_i : \mathcal{T}_i \times Y \to \mathbb{R}$ is the *utility function* of player $i$. We say that a *type* $t_i \in \mathcal{T}_i$ has *utility* $u_i(t_i, a) \in \mathbb{R}$ for an outcome $a \in Y$. A *social choice function* $f$ over $E$ is a mapping $\mathcal{T}_1 \times \dots \times \mathcal{T}_n \to Y$. We restrict attention to deterministic social choice functions.

We say that a social choice function $f : \mathcal{T}_1 \times \dots \times \mathcal{T}_n \to Y$ is *incentive compatible* (or *implementable*) (without transfers) if for any $i$, $t_1 \in \mathcal{T}_1, \dots, t_n \in \mathcal{T}_n$, and $t_i' \in \mathcal{T}_i$, we have

$$u_i(t_i, f(t_i, t_{-i})) \geq u_i(t_i, f(t_i', t_{-i})).$$

That is, each agent (weakly) maximizes their utility by reporting their true type, regardless of the types of other agents.

Our paper works with the paradigm of *monetary transfers and quasilinear utilities*. Unlike many prior papers, we make the distinction between environments with transfers and without transfers explicit. For any environment $E = (Y, \mathcal{T}_1, \dots, \mathcal{T}_n, u_1, \dots, u_n)$, the corresponding *quasilinear environment with transfers* is $E' = (Y \times \mathbb{R}^n, \mathcal{T}_1, \dots, \mathcal{T}_n, u_1', \dots, u_n')$, where $u_i'(t_i, (y, p_1, \dots, p_n)) = u_i(t_i, y) + p_i$. That is, the quasilinear environment adds *transfers* $p_1, \dots, p_n$ to each agent, and the agents quasilinear utility is the sum of its utility for the outcome and the transfer. In this context, we call $u_i(t_i, a)$ the *value* agent $i$ gets (in order to distinguish it from agent $i$'s utility of $u_i(t_i, a) + p_i$). We say that a social choice function $f'$ over $E'$ *computes* a social choice function $f$ over $E$ if $f'$ satisfies $f'(t_1, \dots, t_n) = (f(t_1, \dots, t_n), p_1, \dots, p_n)$ for each $(t_1, \dots, t_n) \in \mathcal{T}_1, \dots, \mathcal{T}_n$. (That is, the function must agree on $Y$, but can be arbitrary on the transfers.)

We treat transfers primarily as a tool for encouraging truthful behavior in mechanisms[17]. Thus, a social choice function $f : \mathcal{T}_1 \times \dots \times \mathcal{T}_n \to Y$ over an environment $E$ is *incentive compatible* (or *implementable*) (with transfers) if there exists a function $f' : \mathcal{T}_1 \times \dots \times \mathcal{T}_n \to Y \times \mathbb{R}^n$ which computes $f$ and is incentive compatible in the quasilinear environment $E'$. This is equivalent to the existence of $n$ transfer functions[18] $p_1, \dots, p_n : \mathcal{T}_1 \times \dots \times \mathcal{T}_n \to \mathbb{R}$ such that

$$u_i(t_i, f(t_i, t_{-i})) + p_i(t_i, t_{-i}) \geq u_i(t_i, f(t_i', t_{-i})) + p_i(t_i', t_{-i}).$$

In this case, we say the transfer functions $(p_i)_{i=1,\dots,n}$ *incentivize* social choice function $f$.

A *mechanism* $\mathcal{M} = (G, S_1, \dots, S_n)$ over an environment $E = (Y, \mathcal{T}_1, \dots, \mathcal{T}_n, u_1, \dots, u_n)$ consists of

- An extensive form game $G$ for $n$ players with perfect recall and consequences in $Y$ (defined below).
- A type-strategy $S_i$ for each player $i$, which maps types $\mathcal{T}_i$ to (behavioural) strategies $s_i$ of player $i$ in game $G$ (defined below).

Mechanism $\mathcal{M}$ over $E$ *computes (without transfers)* a social choice function $f$ over $E$ if we have $G(S_1(t_1), \dots, S_n(t_n)) = f(t_1, \dots, t_n)$ for each profile of types $t_1, \dots, t_n \in \mathcal{T}_1 \times \dots \times \mathcal{T}_n$. A mechanism $\mathcal{M}$ over $E'$ *computes (with transfers)* a social choice $f$ over $E$ if $\mathcal{M}$ computes $f'$, for some $f'$ computing $f$.

We now present our incentive compatibility notions for mechanisms, both with and without transfers (recall that, if the mechanism has transfers, then $u_i(t_i, (y, p_1, \dots, p_n))$ denotes the quasilinear utility $u_i(t_i, y) + p_i$). An implementation $\mathcal{M}$ is *ex-post Nash incentive compatible* (EPIC) if, for any $i$, any types $t_1 \in \mathcal{T}_1, \dots, t_n \in \mathcal{T}_n$, and any behavioral strategy $s_i'$ of player $i$, we have (letting $S_{-i}(t_{-i}) = (S_j(t_j))_{j \neq i}$):

$$u_i(t_i, G(S_i(t_i), S_{-i}(t_{-i}))) \geq u_i(t_i, G(s_i', S_{-i}(t_{-i}))).$$

An implementation is *dominant strategy incentive compatible* (DSIC) if, for any $i$, type $t_i$ of player $i$, behavioral strategies $s_i'$ and $s_{-i}$ of all players,

$$u_i(t_i, G(S_i(t_i), s_{-i})) \geq u_i(t_i, G(s_i', s_{-i})).$$

Thus, ex-post Nash implementations are weaker, as they only require $S_i(t_i)$ to be a best response when other agents are playing strategies consistent with some $S_{-i}(t_{-i})$.

A *direct revelation mechanism* is one in which each agent is asked to simultaneously reveal their type to the mechanism, and then the outcome is computed. In such a mechanism, every possible strategy corresponds to some type, and thus the mechanism is EPIC if and only if it is DSIC.

*Extensive Form Games.* A *deterministic extensive form game with perfect recall and consequences in* $Y$ (hereafter called a *game*) is a tuple $G = (H, E, \mathcal{P}, A, \mathcal{A}, (\mathcal{I}_i)_{i \in [n]}, g)$ such that

- $H$ is a set of states (also called nodes), and $E$ is a set of directed edges between the states, such that $(H, E)$ forms a finite directed tree (where every edge points away from

[17] Alternative paradigms include studying "budget balanced" mechanisms or mechanisms that maximize revenue.

[18] We do not make any assumptions (such as "no positive transfers" or "individual rationality") on the transfers the mechanism is allowed to use. This makes our imposibility results only stronger.

the root). We denote typical elements of $H$ by $h$[19]. Define $Z$ as the set of leafs, and let the root be called $h_0$. Moreover, define $\sigma_E : H \to 2^E$ such that $\sigma_E(h)$ is the set of edges leading out of state $h$ (that is, leading to successor nodes), and $\sigma_H : H \to 2^H$ such that $\sigma(h)$ is the set of states which are immediate successors of $h$ in the game tree. We write edges like $(h, h') \in E$, where $h$ is between the root and $h'$.

- $\mathcal{P} : H \to \{1, \ldots, n\}$ is the player choice function, which labels each non-leaf node in $H \setminus Z$ with the player who acts at that node. Define $H_i := \{s \in S | \mathcal{P}(s) = i\}$ as the set of states where player $i$ is called to act. We assume that no player takes a consecutive turn, that is, for any $h$ and $h' \in \sigma_H(h)$, we have $\mathcal{P}(h) \neq \mathcal{P}(h')$[20].
- $A$ is the set of actions, and $\mathcal{A} : E \to A$ labels each edge with an action. $\mathcal{A}$ must be injective on each set $\sigma_E(h) \subseteq E$ (that is, two edges below the same node cannot be labeled with the same action). Define $A : H \to 2^A$ such that $A(h)$ is the set of actions available at state $h$ (that is, $\{\mathcal{A}(e) | e \in \sigma_E(h)\}$).
- For each player $i$, the set $\mathcal{I}_i$ (the *information partition*) is a partition of $H_i$ (the set of states where $i$ is called to act) such that for every set $I_i \in \mathcal{I}_i$, the set of actions available to $i$ are exactly the same at every set in $I_i$ (that is, $A(s) = A(s')$ for each $s, s' \in I_i$). The elements $I_i \in \mathcal{I}_i$ are called the *information sets* of player $i$. Abusing notation slightly, we denote $A : \mathcal{I}_i \to A$ such that $A(I_i) = A(s)$ for any $s \in N$. When a player acts in $G$, all that player knows is which information set they are in. The information sets must satisfy the following (known as the "perfect recall" assumptions[21]):
  - Any path from the root to a leaf can cross a specific information set only once. That is, for any path $p = (h_0, h_1, \ldots, h_k)$ from the root to a leaf, we never have $h_i, h_j \in I_i \in \mathcal{I}_i$ for $i \neq j$.
  - If two nodes are in the same information set of player $i$, then $i$'s experience in reaching those nodes must be identical. More specifically, for node $h \in H_i$, we define $\psi_i(p)$, the *experience of player $i$ reaching $h$* as follows: take the (unique) path $p = (h_0, h_1, \ldots, h_k = h)$ from the root to $h$, and for each $j \in [k]$ at which $\mathcal{P}(h_j) = i$, write $(I_i^j, a^j)$ in order, where $I_i^j \ni h_j$ is the information set containing $h_j$ and $a^j$ is the (unique) action which player $i$ takes at $h_j$ to move the game to $h_{j+1}$. So $\psi_i(p)$ is an ordered, alternating list of information sets and actions $i$ takes at those information sets. We must have $\psi_i(h) = \psi_i(h')$ for any two histories in the same information set $(h, h' \in I_i \in \mathcal{I}_i)$.
- $g : Z \to Y$ labels each leaf node with an outcome from $Y$.

A *(behavioral) strategy* $s_i$ of player $i$ is a function $\mathcal{I}_i \to A$, such that $s_i(I_i) \in A(I_i)$ is an action available to player $i$ at information set $I_i \in \mathcal{I}_i$. The result of the mechanism under a behavioural strategy profile $(s_1, \ldots, s_n)$ is the outcome in $Y$ in the leaf node which you arrive at by iteratively following the action selected by each $s_i$. We write this as $G(s_1, \ldots, s_n)$. That is, if $(h_0, h_1, \ldots, h_k)$ is the path from the root to a leaf such that each edge $(h_j, h_{j+1})$ is the unique edge such that $\mathcal{A}(h_j, h_{j+1}) = s_i(I_i)$, for $i = \mathcal{P}(h_j)$ and $h_j \in I_i \in \mathcal{I}_i$, then we set $G(s_1, \ldots, s_n) = g(h_k)$.

A game is *perfect information* if every information set is a singleton. Observe that perfect information games cannot hide information from players or allow more than one player to move simultaneously.

For a state $h \in H$, define the communication cost of $h$ as $\lceil \log |A(h)| \rceil$, i.e. the number of bits needed for the agent acting at $h$ to communicate their choice of action. Define the communication cost of game $G$ as the maximum sum of the communication costs of nodes on a path from the root to a leaf node in $G$[22].

Consider an environment $E = (Y, \mathcal{T}_1, \ldots, \mathcal{T}_n, u_1, \ldots, u_n)$ and an extensive form game $G$ with consequences in $Y$. As we mentioned above, a *type-strategy* $S_i$ is a mapping from $\mathcal{T}_i$ to behavioural strategies of player $i$ in game $G$. Equivalently, a type-strategy is any function $S_i : \mathcal{T}_i \times \mathcal{I}_i \to A$ such that $s_i(t_i, I_i) \in A(I_i)$ for each $I_i \in \mathcal{I}_i$. We let $S_i(t_i)$ denote the entire behavioural strategy. For clarity, we capitalize type-strategies. We typically refer to behavioural strategies simply as "strategies", and specify explicitly when $S_i$ is a type-strategy.

*Notation.* When describing games between two players, Alice and Bob, we often use the terms "Alice node" for a node where Alice acts. We also denote such an $h$ with $\mathcal{P}(h) = A$. Similarly, Bob nodes have $\mathcal{P}(h) = B$.

In auction-like domains, we typically identify the type with a valuation function over the bundles of items received by a player. For example, when the allowable types are some sets of functions from subsets of $M$, and the allowable outcomes are partitions of the items to the $n$ players, we formally have $u_i(v_i, (A_1, \ldots, A_n)) = v_i(A_i)$. Thus, we often write $v_i(A_i)$ in place of the entire outcome $(A_1, \ldots, A_n)$.

For a player $i$ with type $t$, we say that $S_i(t)$ is the "truth-telling" strategy of player $i$, and the action $S_i(t)(h)$ is the "truth telling action" at node $h$. A strategy is a *best response* to strategies $(s_j)_{j \neq i}$ for player $i$ with type $t$ if the strategy maximizes player $i$'s utility across all possible strategies of player $i$.

[19] This follows from the economics convention of identifying states with "histories", that is, the (unique) sequence of actions taken to arive in a certain node. We describe the game more concretely in terms of nodes of a tree because in some arguments we need to directly manipulate and change the game tree, which can alter these histories.

[20] Note that this assumption is without loss of generality. Condensing consecutive actions by the same player into a single node leaves the game essentially unchanged. In particular, the communication cost does not increase and if the mechanism was already DSIC, then it is still DSIC.

[21] These assumptions just mean that the mechanism is not able to force agents to forget things they knew in the past of the game. While some of the game theory literature relaxes this assumption, we do not consider games without perfect recall here. This is in accordance with our adversarial model: we consider agents who know the already know the entire game tree in advance (although when they actually act in the mechanism, they only know what the mechanism tells them, that is, which information set they are in).

[22] Some prior works [18] limit mechanisms to at most two actions per node and define the communication cost as the maximum depth of the tree. This is equivalent to our definition up to constants, and our definition allows us to assume without loss of generality that no agent takes consecutive turns in the game.

As is standard in the literature, we do not count the communication which the mechanism must send to the agents (to tell them which information set they are in). In perfect information games, one can argue that this is because the mechanism is simply run over a public communication channel. However, this does not apply in games of partial information. We note that counting the communication the mechanism would need to tell players their information set can increase the communication used by at most a quadratic factor. This is because at worst the mechanism needs to repeat to agent $i$ the messages of all agents which acted before $i$.

## A.1 Describing the Game Tree via Sets of Types

We now give some ways to regularize and describe EPIC mechanisms in a natural way in terms of the types of agents. Most of this language has been considered before (see e.g. [6, 18]).

Let $(G, S_1, \ldots, S_n)$ be a deterministic mechanism EPIC implementing $f$ over environment $E = (Y, \mathcal{T}_1, \ldots, \mathcal{T}_n, u_1, \ldots, u_n)$. For each node $h \in H$, let $\mathcal{T}(h) \subseteq \mathcal{T}_1 \times \ldots \times \mathcal{T}_n$ denote the set of types $(t_1, \ldots, t_n)$ such that, when agents play strategies $(S_1(t_1), \ldots, S_n(t_n))$, the computation of $G(S_1(t_1), \ldots, S_n(t_n))$ enters state $h$ (that is, $h$ is on the path from the root to a leaf taken when computing $G(S_1(t_1), \ldots, S_n(t_n))$).

Lemma A.1. *Each set $\mathcal{T}(h)$ is a rectangle. That is, $\mathcal{T}(h) = \mathcal{T}_1(h) \times \ldots \times \mathcal{T}_n(h)$ for some sets $\mathcal{T}_1(h), \ldots, \mathcal{T}_n(h)$. Moreover, if $\mathcal{P}(h) = i$ then $\{\mathcal{T}_i(h')\}_{h' \in \sigma_H(h)}$ is a partition of $\mathcal{T}_i(h)$. If $\mathcal{P}(h) \neq i$ then $\mathcal{T}_i(h') = \mathcal{T}_i(h)$ for all $h' \in \sigma_H(h)$.*

Proof. One can apply a standard rectangle argument to $\mathcal{T}(h)$. Specifically, suppose that $t = (t_1, \ldots, t_n), (t'_1, \ldots, t'_n) \in \mathcal{T}(h)$, and consider some player $i$ and type profile $(t'_i, t_{-i})$. Every player other than $i$ takes the same action under $t$ and $(t'_i, t_{-i})$. Because the actions taken by each player along the path from the root to $h$ are unique, $S_i(t'_i)$ must play the same actions along this path as does $S_i(t_i)$. Thus, $i$ will take the same action under $t_i$ and $t'_i$ at every node along the path where $i$ is called to act. So $(t'_i, t_{-i}) \in \mathcal{T}(h)$ as well. Applying this for each player $i = 1, \ldots, n$ proves that $\mathcal{T}(h)$ is a rectangle. This makes $\mathcal{T}_1(h), \ldots, \mathcal{T}_n(h)$ well defined for each node $h$.

If a player does not act at node $h$, then each successor of node $h$ keeps the type set of that player the same, by definition. On the other hand, when player $i$ acts at $h$, every type in $\mathcal{T}_i(h)$ takes exactly one action at $h$. This proves $\mathcal{T}_i(h')$ partitions $\mathcal{T}_i(h)$ as we let $h' \in \sigma_H(h)$ vary. □

Without loss of generality, we may assume that every node $h$ of the game has $\mathcal{T}(h) \neq \emptyset$[23]. Under this assumption, to specify an EPIC mechanism, it suffices to specify a tree equipped with type sets $\mathcal{T}_i(h)$ satisfying the conclusions of lemma A.1. In appendix B in the full version, we often describe modifying extensive form games in these terms.

[23] Removing nodes for which this is true can only decrease the communication complexity, while still computing the correct result in dominant strategies (if the original mechanism did so).

## REFERENCES


[1] Itai Ashlagi and Yannai A Gonczarowski. 2018. Stable matching mechanisms are not obviously strategy-proof. *Journal of Economic Theory* 177 (2018), 405–425.

[2] Sepehr Assadi, Hrishikesh Khandeparkar, Raghuvansh R. Saxena, and S. Matthew Weinberg. 2020. Separating the communication complexity of truthful and non-truthful combinatorial auctions. In *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing, STOC 2020, Chicago, IL, USA, June 22-26, 2020*, Konstantin Makarychev, Yury Makarychev, Madhur Tulsiani, Gautam Kamath, and Julia Chuzhoy (Eds.). ACM, 1073–1085. https://doi.org/10.1145/3357713.3384267

[3] Sepehr Assadi and Sahil Singla. 2019. Exponentially Improved Truthful Combinatorial Auctions with Submodular Bidders. In *Proceedings of the Sixtieth Annual IEEE Foundations of Computer Science (FOCS)*.

[4] Moshe Babaioff, Liad Blumrosen, and Michael Schapira. 2013. The communication burden of payment determination. *Games and Economic Behavior* 77, 1 (2013), 153 – 167. https://doi.org/10.1016/j.geb.2012.08.007

[5] Moshe Babaioff, Robert D. Kleinberg, and Aleksandrs Slivkins. 2015. Truthful Mechanisms with Implicit Payment Computation. *J. ACM* 62, 2, Article 10 (May 2015), 37 pages. https://doi.org/10.1145/2724705

[6] Sophie Bade and Yannai A. Gonczarowski. 2017. Gibbard-Satterthwaite Success Stories and Obvious Strategyproofness. In *Proceedings of the 2017 ACM Conference on Economics and Computation* (Cambridge, Massachusetts, USA) *(EC '17)*. Association for Computing Machinery, New York, NY, USA, 565. https://doi.org/10.1145/3033274.3085104

[7] Mark Braverman, Jieming Mao, and S. Matthew Weinberg. 2018. On Simultaneous Two-player Combinatorial Auctions. In *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2018, New Orleans, LA, USA, January 7-10, 2018*. 2256–2273. https://doi.org/10.1137/1.9781611975031.146

[8] David Buchfuhrer, Shaddin Dughmi, Hu Fu, Robert Kleinberg, Elchanan Mossel, Christos H. Papadimitriou, Michael Schapira, Yaron Singer, and Christopher Umans. 2010. Inapproximability for VCG-Based Combinatorial Auctions. In *Proceedings of the Twenty-First Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*.

[9] Amit Daniely, Michael Schapira, and Gal Shahaf. 2015. Inapproximability of Truthful Mechanisms via Generalizations of the VC Dimension. In *Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing, STOC 2015, Portland, OR, USA, June 14-17, 2015*. 401–408. https://doi.org/10.1145/2746539.2746597

[10] Shahar Dobzinski. 2007. Two Randomized Mechanisms for Combinatorial Auctions. In *Proceedings of the 10th International Workshop on Approximation and the 11th International Workshop on Randomization, and Combinatorial Optimization. Algorithms and Techniques*. 89–103.

[11] Shahar Dobzinski. 2016. Computational Efficiency Requires Simple Taxation. In *FOCS*.

[12] Shahar Dobzinski and Noam Nisan. 2011. Limitations of VCG-based mechanisms. *Combinatorica* 31, 4 (2011), 379–396. https://doi.org/10.1007/s00493-011-2528-4

[13] Shahar Dobzinski, Noam Nisan, and Michael Schapira. 2010. Approximation Algorithms for Combinatorial Auctions with Complement-Free Bidders. *Math. Oper. Res.* 35, 1 (2010), 1–13. https://doi.org/10.1287/moor.1090.0436

[14] Shahar Dobzinski and Shiri Ron. 2020. The Communication Complexity of Payment Computation. *Manuscript* (2020).

[15] Shahar Dobzinski and Michael Schapira. 2006. An Improved Approximation Algorithm for Combinatorial Auctions with Submodular Bidders. In *Proceedings of the Seventeenth Annual ACM-SIAM Symposium on Discrete Algorithm* (Miami, Florida) *(SODA '06)*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 1064–1073. http://dl.acm.org/citation.cfm?id=1109557.1109675

[16] Shahar Dobzinski and Jan Vondrák. 2013. Communication Complexity of Combinatorial Auctions with Submodular Valuations. In *Proceedings of the Twenty-Fourth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2013, New Orleans, Louisiana, USA, January 6-8, 2013*, Sanjeev Khanna (Ed.). SIAM, 1205–1215. https://doi.org/10.1137/1.9781611973105.87

[17] Tomer Ezra, Michal Feldman, Eric Neyman, Inbal Talgam-Cohen, and S. Matthew Weinberg. 2019. Settling the Communication Complexity of Combinatorial Auctions with Two Subadditive Buyers. In *the 60th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*.

[18] Ronald Fadel and Ilya Segal. 2009. The communication cost of selfishness. *Journal of Economic Theory* 144, 5 (2009), 1895–1920.

[19] Uriel Feige. 2009. On Maximizing Welfare When Utility Functions Are Subadditive. *SIAM J. Comput.* 39, 1 (2009), 122–142. https://doi.org/10.1137/070680977

[20] Uriel Feige and Jan Vondrák. 2010. The Submodular Welfare Problem with Demand Queries. *Theory of Computing* 6, 1 (2010), 247–290. https://doi.org/10.4086/toc.2010.v006a011

[21] Eyal Kushilevitz and Noam Nisan. 1997. *Communication complexity*. Cambridge University Press.

[22] Ron Lavi, Ahuva Mu'alem, and Noam Nisan. 2003. Towards a Characterization of Truthful Combinatorial Auctions. In *44th Symposium on Foundations of Computer Science (FOCS 2003), 11-14 October 2003, Cambridge, MA, USA, Proceedings*. 574–583. https://doi.org/10.1109/SFCS.2003.1238230

[23] Ron Lavi and Chaitanya Swamy. 2005. Truthful and Near-Optimal Mechanism Design via Linear Programming. In *Proceedings of the 46th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*.

[24] Daniel Lehmann, Liadan O'Callaghan, and Yoav Shoham. 2002. Truth revelation in approximately efficient combinatorial auctions. *J. ACM* 49, 5 (2002), 577–602. https://doi.org/10.1145/585265.585266

[25] Shengwu Li. 2017. Obviously strategy-proof mechanisms. *American Economic Review* 107, 11 (2017), 3257–87.

[26] Noam Nisan and Ilya Segal. 2006. The communication requirements of efficient allocations and supporting prices. *J. Economic Theory* 129, 1 (2006), 192–224. https://doi.org/10.1016/j.jet.2004.10.007

[27] Christos H. Papadimitriou, Michael Schapira, and Yaron Singer. 2008. On the Hardness of Being Truthful. In *Proceedings of the 49th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*.

[28] Marek Pycia and Peter Troyan. 2019. Obvious Dominance and Random Priority. In *Proceedings of the 2019 ACM Conference on Economics and Computation, EC 2019, Phoenix, AZ, USA, June 24-28, 2019*. 1. https://doi.org/10.1145/3328526.3329613