The Revelation Principle for Mechanism Design with Reporting Costs

ANDREW KEPHART, Duke University

VINCENT CONITZER, Duke University

The *revelation principle* is a key tool in mechanism design. It allows the designer to restrict attention to the class of truthful mechanisms, greatly facilitating analysis. This is also borne out in an algorithmic sense, allowing certain computational problems in mechanism design to be solved in polynomial time. Unfortunately, when not every type can misreport every other type (the *partial verification* model), or—more generally—misreporting can be costly, the revelation principle can fail to hold. This also leads to NP-hardness results. The primary contribution of this paper consists of characterizations of conditions under which the revelation principle still holds when misreporting can be costly. (These are generalizations of conditions given earlier for the partial verification case [Green and Laffont 1986; Yu 2011].) We also study associated computational problems.

Additional Key Words and Phrases: automated mechanism design, signaling costs, partial verification, revelation principle

1. INTRODUCTION

Mechanism design concerns making decisions based on the private information of one or more agents, who will not report this information truthfully if they do not see this to be in their interest. The goal is to define a *mechanism*—a game to be played by the agent(s)—that defines actions for the agents to take and maps these actions to outcomes, such that in equilibrium, the agents' true types map to outcomes as desired. This may, at first, appear to leave us in the unmanageable situation of having to search through all possible games we could define. Fortunately, there is the *revelation principle*, which states that anything that can be implemented by a mechanism can also be implemented by a *truthful* mechanism, where agents in equilibrium report their types truthfully. This vastly reduces the search space, and indeed under certain circumstances allows optimal mechanisms to be found in polynomial time (e.g., if the number of agents is constant and randomized outcomes are allowed [Conitzer and Sandholm 2002, 2004]). And of course, the revelation principle is also extremely useful in the process of obtaining analytical results in mechanism design.

Authors' addresses: Andrew Kephart, Department of Computer Science, Duke University; email: kephart@cs.duke.edu; Vincent Conitzer, Department of Computer Science, Duke University; email: conitzer@cs.duke.edu

EC'16, July 24–28, 2016, Maastricht, The Netherlands. ACM 978-1-4503-3936-0/16/07 ...$15.00.

http://dx.doi.org/10.1145/2940716.2940795

Unfortunately[1], there are situations where the revelation principle fails to hold. Notably, this is the case in settings with so-called *partial verification*. This means that not every type is able to report every other type (without detection). For example, an agent in an online marketplace may be able to pretend to arrive later than she really does, but not earlier. Indeed, due to this absence of a revelation principle in this context, the problem of determining whether a particular choice function can be implemented becomes, in general, NP-hard [Auletta et al. 2011]. It is thus natural to ask the question under which conditions on the misreporting graph—which represents which types can misreport which other types—the revelation principle holds. Such a characterization can help us both obtain analytical results in mechanism design as well as algorithms for efficiently finding mechanisms. Indeed, these conditions have been previously characterized [Green and Laffont 1986; Yu 2011]; we review them later as they come up as special cases of our characterizations.

Nevertheless, in practice, it is not always black and white whether one type can misreport another. Often, one type can misreport another *at some cost*. This cost may correspond to financial cost or effort. For example, a minor may acquire a fake driver's license at some cost. A consumer may engage in tricks to improve his credit score. In college admissions, a student can improve on his/her "natural" SAT score by extra prepping for the test.[2] And so on. This falls in the general domain of signaling games, such as Spence's famous model of education [Spence 1973]. Generally, for every type θ and every type $\hat{\theta}$ she might misreport, there is a cost $c(\theta, \hat{\theta})$ for doing so. Traditional mechanism design is the case where $c(\theta, \hat{\theta}) = 0$ everywhere; partial verification is the special case where $c(\theta, \hat{\theta}) \in \{0, \infty\}$.

In this paper, we characterize when the revelation principle holds with arbitrary misreporting costs. The previous characterizations for partial verification straightforwardly follows as a special case. We also consider cases where some aspects beyond the misreporting costs, such as the valuation function, are known.

1.1. Related Work

In earlier work, we studied the computational complexity of deciding whether a given choice function can be implemented when misreporting is costly [Kephart and Conitzer 2015]; these results will be relevant in Section 6 of this paper.

[Kartik and Tercieux 2012] gives a necessary condition, *evidence-monotonicity*, which is required for a social choice function to be implementable with no signaling costs incurred in equilibrium. Applications of signaling costs to more specific settings include [Bull 2008], [Lacker and Weinberg 1989], [Kartik 2008], and [Deneckere and Severinov 2007]. In the latter it is shown that if the agent can send multiple signals, in the limit as the number of these increases the principal can elicit the agent's private information at a very small cost.

A different variant that also does not have the black-and-white nature of partial verification is that of *probabilistic verification*, in which a lying agent is caught with some *probability* [Caragiannis et al. 2012].

[1]This is only unfortunate from analytical and algorithmic viewpoints. Indeed, often a non-truthful mechanism in these settings will perform better than any truthful one. We show several examples of this.

[2]We have in mind here not prepping that has value beyond the test—e.g., studying algebra in order to be able to solve more problems—but rather acquiring tricks—e.g., about how best to guess when unsure—that improve the score on the test but are otherwise of no societal value.

We also consider our work related to machine learning in contexts where what is being classified is an agent that may put in some effort to resist accurate classification. The most obvious version of this is *adversarial classification*: detecting spam, intrusions, fraud, etc. when the perpetrators wish to evade detection [Barreno et al. 2010; Dalvi et al. 2004].

However, as the examples in the introduction indicate, there are many situations which are not zero-sum. This is also brought out in more recent work on "strategic classification," which takes less of a mechanism design viewpoint than we do [Hardt et al. 2016].

1.2. Introductory Example - Inspection Game

We now provide an introductory example that illustrates our model as well as the failure, in general, of the revelation principle when misreporting is costly.[3] We will give another, more complex example in Section 3, which we will use as a running example throughout the paper.

Suppose Beth is considering buying a crate of produce from Pam. The produce can be fresh, decent, or rotten (and Pam will know which it is). Beth can either accept or reject the crate. If the produce is fresh or decent, Beth would like to accept it, otherwise she would like to reject it. This gives:

$\Theta = \{\textit{fresh}, \textit{decent}, \textit{rotten}\}$ — the set of types.
$O = \{\textit{accept}, \textit{reject}\}$ – the set of outcomes.
$F = \{\textit{fresh} \rightarrow \textit{accept}, \textit{decent} \rightarrow \textit{accept}, \textit{rotten} \rightarrow \textit{reject}\}$ – the choice function Beth seeks to implement.

Before making her decision, Beth can inspect the appearance of the produce. However, at some cost Pam can add dyes or scents to the produce, which will alter how it appears to Beth. The following matrix gives the cost of making a crate of type θ appear to be of type $\hat{\theta}$. For example, at a cost of 30, Pam can make a crate of rotten produce appear fresh.

$c(\theta, \hat{\theta}) =$

	$\widehat{\textit{fresh}}$	$\widehat{\textit{decent}}$	$\widehat{\textit{rotten}}$
fresh	0	0	0
decent	10	0	0
rotten	30	10	0

The value that Pam receives if the crate is accepted is 20:[4]

$v_\theta(\textit{accept}) = 20$
$v_\theta(\textit{reject}) = 0$

Beth needs to commit to a mechanism for choosing an outcome based on how the produce appears. The naïve mechanism of accepting the produce whenever it does not appear rotten, i.e., $A = \{\widehat{\textit{fresh}} \rightarrow \textit{accept}, \widehat{\textit{decent}} \rightarrow \textit{accept}, \widehat{\textit{rotten}} \rightarrow \textit{reject}\}$ would fail. It would be worth it to Pam to pay the cost of 10 to make rotten produce appear to be decent, netting 10 value. Hence Beth would end up inadvertently accepting rotten produce.

[3]We present a similar example in [Kephart and Conitzer 2015]. This is the smallest example where a choice function is implemented by some nontruthful mechanism, but not by any truthful one.
[4]In general, the value of each outcome to the agent can depend on the type.

What Beth should do instead is use $A^* = \{\hat{fresh} \rightarrow accept, \hat{decent} \rightarrow reject, \hat{rotten} \rightarrow reject\}$. Under A^*, if the produce really is rotten it will not be worth it for Pam to alter it, and it will be rejected. On the other hand, if the produce is decent, Pam will make it appear to be fresh and it will be accepted, resulting in the implementation of Beth's desired choice function.

2. MODEL

As is common in this type of setting, we focus on the case of a single type-reporting agent; this corresponds to holding the other agents' reports fixed.

In a fully specified instance, we have a set of *types* Θ; we will generally use $\theta, \theta_1, \theta_2, \ldots$ to denote variable types and $a, b, c, \ldots$ to denote specific types. We also have a set of *outcomes* (alternatives) O. There is a *valuation function* $v : \Theta \times O \rightarrow \mathbb{R}$, where $v(\theta, o)$ is the valuation that type θ has for outcome o. We sometimes write $v_\theta(o)$ rather than $v(\theta, o)$. Finally, there is a *cost function* $c : \Theta \times \Theta \rightarrow \mathbb{R}_{\geq 0}$, where $c(\theta_1, \theta_2)$ denotes the cost type θ_1 incurs when misreporting θ_2. Reporting truthfully is always costless. We sometimes use the shorthand ab for $c(a, b)$.

A *(direct-revelation) mechanism* is defined by, first, an *allocation function* $A : \Theta \rightarrow O$, where $A(\hat{\theta}) = o$ denotes that the mechanism chooses o when type $\hat{\theta}$ is reported. (When there is a risk of confusion, we generally use θ to denote the true type and $\hat{\theta}$ to denote the reported type. Note that the fact that the mechanism is a direct-revelation mechanism—i.e., agents report types directly—does not mean it is necessarily truthful.) When we allow for transfers, then another part of the mechanism is the *transfer function* $T : \Theta \rightarrow \mathbb{R}$, where $T(\hat{\theta})$ denotes the transfer (payment) *received* by the agent when reporting $\hat{\theta}$. (Hence, $T(\hat{\theta}) < 0$ implies the agent is making a transfer.) In a mechanism without transfers, $T(\hat{\theta}) = 0$ for all $\hat{\theta}$. The agent's utility for having type θ, reporting type θ', receiving outcome o and transfer t is $u(\theta, \theta', o, t) = v_\theta(o) - c(\theta, \theta') + t$.

We are generally interested in *implementing* a *choice function* $F : \Theta \rightarrow O$. We sometimes use the shorthand o_a for $F(a)$. Let $S : \Theta \rightarrow \Theta$ denote a *strategy* for the agent, where $S(\theta) = \hat{\theta}$ denotes that the agent reports $\hat{\theta}$ when her true type is θ. We say S is *optimal* for mechanism $M = (A, T)$ if for all θ and $\hat{\theta}$, $u(\theta, S(\theta), A(S(\theta)), T(S(\theta))) \geq u(\theta, \hat{\theta}, A(\hat{\theta}), T(\hat{\theta}))$.

A mechanism $M = (A, T)$ together with strategy S *implements* F if S is optimal for M and for all θ, $A(S(\theta)) = F(\theta)$. (Moreover, it implements it with transfer $T(S(\theta))$ and utility $u(\theta, S(\theta), A(S(\theta)), T(S(\theta)))$ for type θ.)

We will sometimes use N to denote a not-necessarily truthful mechanism and H to denote a truthful one.[5] Additionally, S_N and S_H will refer to optimal strategies for the respective mechanisms. We use $U_M(\theta)$ to refer to the utility that an agent of type θ achieves by reporting optimally under M. That is, $U_M(\theta) = u(\theta, S_M(\theta), A_M(S_M(\theta)), T_M(S_M(\theta)))$.

2.1. Revelation Principle

We say that a revelation principle (RP) holds whenever we can, without loss of generality, restrict our attention to truthful mechanisms when trying to implement the choice

[5]Here, H stands for "honest" to prevent confusion with the transfer function T.

function. Thus, we say that given Θ and c (we will refer to a combination of Θ and c as an *instance*), the *revelation principle holds* if for any O, v, $N = (A_N, T_N)$, and S_N which is optimal for N, there exists another mechanism $H = (A_H, T_H)$ such that the truthful-reporting strategy (identity function) S_H (with $S_H(\theta) = \theta$) is optimal for H, and for all θ, $A_H(\theta) = A_N(S_N(\theta))$. Hence, any choice function F that can be implemented can be implemented truthfully.

Sometimes, we will also wish to implement the choice function with specific transfers and/or utilities. We say that the revelation principle holds with *fixed transfers* if the mechanism H can always be chosen such that $T_H(\theta) = T_N(S_N(\theta))$ for all θ, and with *fixed utilities* if the mechanism H can always be chosen such that $u(\theta, \theta, A_H(\theta), T_H(\theta)) = u(\theta, S_N(\theta), A_N(S_N(\theta)), T_N(S_N(\theta)))$ for all θ. Note that if the revelation principle holds with fixed transfers, then it also holds when no transfers are possible at all, since this is simply the special case where the transfers are fixed to 0.

3. RUNNING EXAMPLE: STOCKING FOOD BANKS

Throughout the paper we will use a running example of stocking food banks to help illustrate the various instantiations of our framework. (This example is purely for illustration purposes.) Imagine that a city has four districts: *North*, *East*, *South*, *West*, each of which has a food bank and a population of people living in it. A person's type thus consists of the location in which she lives.

Based on demographics and health conditions, the city government has determined how much it values the population of each district receiving certain types of food. For example, it may wish to distribute milk in a district with many families with young children, and vegetables in a district with many single, middle-aged people. Note that the city's objective here is different from maximizing the sum of the utilities of the people who will make use of the food bank. For example, the latter may solely want to consume tasty food, whereas the city may prefer for them to eat healthy food in order to reduce the burden on the health care system. For the purpose of our example, we assume there are three food types, namely (those high in) *fiber*, *protein*, and *vitamins*.

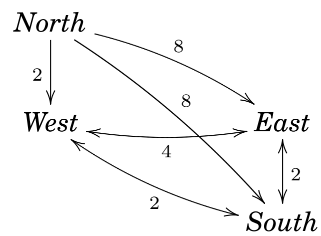
Determining which food to stock in each bank would be straightforward except that there is a possibility that the population from one district might travel to another district if it prefers the latter district's food. This would correspond to them "lying" about their district. We assume here that the food banks cannot check the home address of people arriving at them.

That being said, such "misreporting" is not without cost, because it requires traveling to another district. The transportation costs to residents for traveling between districts are summarized as follows:

$c(\theta, \hat{\theta}) =$

	North	*West*	*East*	*South*
North	0	2	8	8
West	∞	0	4	2
East	∞	4	0	2
South	∞	2	2	0

It can also be useful to visualize this reporting cost structure as a graph. The directed edge between two districts represents the cost of traveling from one to the other. We leave off the zero-cost self reporting edges. All other unshown edges are assumed to have infinite cost.



Suppose that the people of each district value the food as follows.

$v(\theta, o) =$

	fiber	*protein*	*vitamins*
North	1	9	1
West	2	3	3
East	1	3	9
South	1	1	1

Moreover, suppose that the city's objective function is as follows.

$J(\theta, o) =$

	fiber	*protein*	*vitamins*
North	10	0	0
West	0	10	5
East	0	10	5
South	0	5	10

The city's problem is to specify a mechanism—that is, which food each district's food bank distributes—such that it is happy with the equilibrium result of this mechanism. Possibly, the city also has the option to distribute a (welfare) transfer to each person who comes to the food bank.

We assume that there are no capacity constraints on any food bank—that is, the food bank does not run out of food if too many people come to it. Hence, this is effectively a mechanism design problem for a single agent—the person turning up at the food bank—because other agents' decisions do not affect this agent.

There are multiple variants of this problem, depending on whether the city can make transfers, whether it wants these transfers to be a certain amount, whether it cares about the transportation costs incurred by people traveling to a different district, etc.

If the revelation principle holds for the variant in question, the city's problem is significantly easier; it can focus on truthful implementations, i.e., mechanisms such that nobody would travel to another district. On the other hand, if it does not hold, the city needs to consider mechanisms that do incentivize some districts' populations to travel to the food bank in another district, because this may result in better outcomes than any truthful mechanism can achieve. We will see examples of both cases.

4. RESULTS

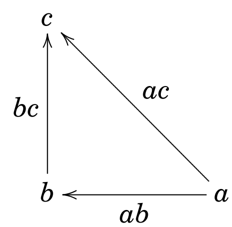
We now move on to our contributions.

4.1. High Level Overview

We first provide a high level overview of the intuition behind our main results.

The revelation principle holds iff for any choice function F implemented by a non-truthful mechanism N, there also exists a truthful mechanism H which implements F. For the purpose of providing intuition, let us limit ourselves here to the case where there is only one non-truthful report in N. Under N let b obtain $F(b)$ by reporting some other type c.

To create H where $A(b) = F(b)$, b must now obtain $F(b)$ by reporting b. This risks that some type a now prefers reporting b to truthfully reporting a. We can visualize these types and the relevant reporting costs between them as follows[6]:



There are two ways to keep a truthful in H.

First: Pay an agent extra for reporting a. This keeps a honest but may cause other types to misreport under H. So we pay off all the types which can report to a too, and so on. This works as long as the process terminates without reaching b. If we had to pay b, the extra transfers would cancel out and all would be for naught. Thus, with variable transfers and variable utilities, if there exists no path of finite reporting cost edges from b to a the revelation principle holds.

Second: Pay less to an agent for reporting b. We can safely subtract bc[7] from the transfer that b received for reporting c under N (making b equally happy under H and N). Note that a did not report c under N. Consider how much greater a's incentive to report b under H is than to report c under N. It is $ac - ab - bc$, which will be nonpositive iff $ac \leq ab + bc$, i.e. the triangle inequality holds among a, b, and c, in which case a still will not misreport. Thus, with variable transfers if the triangle inequality holds the revelation principle holds with fixed utilities (and thus with variable utilities as well).

If we have fixed transfers, neither of the previous two techniques are allowed. So a must prefer not to report b without any altering of transfers. This could only be the case if it is no easier for a to report b than it could c, i.e. $ab \geq ac$.

Finally, with fixed transfers and fixed utilities b must receive the same utility in both H and N. Clearly this can only be the case if $bc = 0$. If so, this reduces to the partial verification setting and we can use the known revelation principle for that case.

4.2. Variable Transfers, Variable Utilities

Consider the case where we allow the transfers and utilities agents achieve to vary between non-truthful and truthful implementations. That is, all that is needed for the revelation principle to hold is that for any non-truthful implementation of a choice function, there exists a truthful one as well—but the transfers and utilities received by types may be different in the latter.

[6]Other types not shown in the graph may exist.
[7]Subtracting any more might cause b to misreport under H.

Definition 4.1 (*VTVU condition*). An instance satisfies the *VTVU condition* if for all ordered triples a, b, c of types, either:

(i) There exists no path of finite reporting cost edges from b to a, or

(ii) $ac \leq ab + bc$

i.e., the triangle inequality holds on the reporting cost structure.

THEOREM 4.2. *The RP holds with variable transfers and variable utilities iff the VTVU condition holds.*

PROOF.

VTVU condition $\implies$ RP holds with variable transfers and variable utilities. Consider the reporting graph consisting of only the edges with finite reporting costs. Consider the strongly connected components of this graph; the graph decomposes into a DAG over these components

Suppose there is a mechanism N that, together with a strategy S_N, nontruthfully implements a choice function F. Let us restrict our attention to a single component. Within it, the triangle inequality must hold since there clearly is a path between every pair of types in it. Hence, within this component the RP holds even with variable transfers and fixed utilities (Theorem 4.5), and hence there exists a truthful implementation within this component.

Choose such an internally truthful implementation (including transfers) within each component; this does not necessarily result in a mechanism that is truthful overall, because there may be incentives to misreport across components. We can fix this as follows.

We order the components in a way consistent with the DAG, such that types in an earlier component can report types in a later component, but not vice versa. Additionally, for each component we specify an additional transfer that all types in that component will receive. By making this transfer sufficiently larger for earlier components in this order, no type will wish to misreport to a later component (and misreporting to an earlier component comes at infinite cost).

(Specifically, if component C_1 comes before C_2, then the additional transfers $\pi_{C_1}^{\text{add}}$ and $\pi_{C_2}^{\text{add}}$ should be such that for all $\theta_1 \in C_1$ and $\theta_2 \in C_2$,

$$v_{\theta_1}(F(\theta_1)) + \pi_{\theta_1}^{\text{orig}} + \pi_{C_1}^{\text{add}} \geq v_{\theta_1}(F(\theta_2)) + \pi_{\theta_2}^{\text{orig}} + \pi_{C_2}^{\text{add}} - \theta_1\theta_2$$

$$\Leftrightarrow$$

$$\pi_{C_1}^{\text{add}} - \pi_{C_2}^{\text{add}} \geq v_{\theta_1}(F(\theta_2)) + \pi_{\theta_2}^{\text{orig}} - \theta_1\theta_2 - v_{\theta_1}(F(\theta_1)) - \pi_{\theta_1}^{\text{orig}}$$

where the π^{orig} are the transfers obtained from applying the VTFU revelation principle within the components.)

Since the additional transfer to a component is the same for all types in it, internal truthfulness is maintained. So, since no type will misreport within a component, or now to another component, the implementation is truthful.

VTVU not holding $\implies$ RP does not hold with variable transfers and variable utilities. Given that the VTVU condition does not hold, we must have some a, b, c with $ac > ab + bc$ for which there is a path of finite-cost edges $b = \theta_1, \theta_2, \ldots, \theta_{l-1}, \theta_l = a$ (where we consider $l+1 = 1$). If the path goes through c, we can assume without loss of generality that $c = \theta_2$.

Assume first $c = \theta_2$.

Consider an outcome set with a separate outcome o_{θ_i} for every type θ_i (except $o_b = o_c$), and a valuation function with $v_{\theta_i}(o_{\theta_i}) = c(\theta_i, \theta_{i+1})$, $v_{\theta_i}(o_{\theta_{i+1}}) = 2c(\theta_i, \theta_{i+1})$ (except $v_a(o_b) = ab + (ac + ab + bc)/2$ and $v_b(o_c) = v_b(o_b) = bc$), and $v = 0$ elsewhere.

Consider the mechanism N where $A_N(\theta_i) = o_{\theta_i}$, except $A_N(b) = o_a$; moreover, let T_N be a large constant value on all the θ_i and c, and 0 everywhere else, so that none of the θ_i would misreport somewhere outside that set. Then an optimal strategy has $S_N(\theta_i) = \theta_i$, except $S_N(b) = c$. This is because for any $\theta_i \notin \{a, b\}$, the only viable alternative is to misreport θ_{i+1}, but the cost of doing so exactly cancels out the benefit; for b, misreporting c results in utility $bc - bc = 0$, which is just as good as reporting truthfully and getting o_a; for a, misreporting c would give utility $T_N + ab + (ac + ab + bc)/2 - ac$, which is less than the $T_N + ab$ it gets for reporting truthfully.

Hence, this is a non-truthful implementation of some choice function with $F(\theta_i) = o_{\theta_i}$.

On the other hand, such a choice function cannot be implemented truthfully. This is because we have a Rochet-style negative cycle [Rochet 1987]. For truthful reporting we must have $T_H(\theta_i) \geq T_H(\theta_{i+1})$ for each θ_i, except $T_H(b) \geq T_H(c) - bc$ and $T_H(a) \geq T_H(b) + (ac + ab + bc)/2 - ab > T_H(b) + ab + bc - ab = T_H(b) + bc$. Hence $T_H(a) > T_H(c)$. But this leads to a contradiction as we follow the inequalities around the cycle.

For the case where $c \neq \theta_2$, we can make a similar argument, treating c as before but using a separate outcome o_{θ_2} for θ_2 and letting $v_b(o_{\theta_2}) = b\theta_2$.

By doing so, b is indifferent between c and θ_2 in the non-truthful implementation which therefore still works[8], and for the cycle we get $T_H(b) \geq T_H(\theta_2) - bc$, allowing us to get the same contradiction. $\square$

4.2.1. Relation to Partial Verification. In the partial verification case (reporting costs are zero or infinity), if we allow transfers to vary, it is known that the revelation principle is characterized by the following “Strong Decomposability” condition [Yu 2011]: in the reporting graph (where there is an edge from a to b if and only if a can report b, at zero cost), (1) every strongly connected component is fully connected, i.e., every type in the component can report every other type; and (2) within a strongly connected component, all types have the same image set, i.e., the set of types they can report is the same.

PROPOSITION 4.3. *In the partial verification case (reporting costs are zero or infinity), strong decomposability is equivalent to the VTVU condition.*

Of course, if both results are correct, this must in fact be the case. Nevertheless, it is instructive to verify it directly.

PROOF.

The VTVU condition $\implies$ Strong Decomposability. First, to show (1), assume a, b, c are in the same strongly connected component (so in particular, a is reachable from

[8]So b still reports to c.

b with edges of zero reporting cost), and that $ab = 0$ and $bc = 0$. Then, by VTVU, $ac \le ab + bc = 0$, so $ac = 0$. This implies transitivity, and hence full connectivity, within every strongly connected component. Second, to show (2), suppose a and b are in the same strongly connected component—so by (1), $ab = 0$—and $bc = 0$. Because a is reachable from b, by VTVU $ac \le ab + bc = 0$, so $ac = 0$. Hence, nodes in a strongly connected component have the same image set.

Strong Decomposability $\implies$ the VTVU condition. Suppose $ab = bc = 0$ and a is reachable from b with edges of zero reporting cost; it suffices to show that $ac = 0$. But in fact, because a and b are in the same strongly connected component, by (2), $ac = 0 \implies bc = 0$.[9] □

4.2.2. Revisiting the Running Example. Suppose the city wants to maximize its objective without regard to the transfers it makes or the resulting utilities of the different types of agent. When we consider the conditions for the revelation principle we just obtained, we see that the triangle inequality is violated by the following triples: (*North, West, East*) and (*North, West, South*). However, in both cases there is no path back to *North*.

Hence, in fact the revelation principle holds. Thus, any choice function that is implementable is also implementable truthfully. So the city only needs to search through the space of truthful mechanisms to maximize its objective. In fact, the following truthful mechanism achieves the best conceivable objective value of 40.

$H =$

	$\hat{North}$	$\hat{West}$	$\hat{East}$	$\hat{South}$
A	*fiber*	*protein*	*protein*	*vitamins*
T	6	0	4	0

4.3. Variable Transfers, Fixed Utilities

We now consider the case where, given a non-truthful mechanism, we wish to obtain a truthful mechanism that implements the same choice function and maintains the utility that each type obtains, but we allow the two mechanisms to be different in the transfers they make to the agents.

Definition 4.4 (*VTFU condition*). An instance satisfies the *VTFU condition* if for all ordered triples a, b, c of types,

$$ac \le ab + bc$$

i.e., the triangle inequality holds on the reporting cost structure.

THEOREM 4.5. *The RP holds with variable transfers and fixed utilities iff the VTFU condition holds.*

PROOF.

VTFU condition $\implies$ RP holds with variable transfers and fixed utilities. By way of contradiction, assume there exists an instance where the VTFU condition holds, but the RP does not. That is, there exists a mechanism N together with a strategy S_N, such that the mechanism H given by $A_H = A_N \circ S_N$ and for all θ,

$$T_H(\theta) = T_N(S_N(\theta)) - \theta S_N(\theta)$$

[9]The careful reader may wonder why we did not need to use condition (1) in this part of the proof. This is because condition (1) in Strong Decomposability is in fact redundant—condition (2) implies condition (1).

(which are the only transfers that could result in θ getting the same utility under both N and H) does not truthfully implement F.

Given that H is not truthful, there must exist two distinct types a, b with $o_b = A_N(S_N(b))$ and $o_a = A_N(S_N(a))$ such that under H, a prefers to misreport to b, hence:

$$U_H(a) = v_a(o_b) + T_H(b) - ab > v_a(o_a) + T_H(a) = U_N(a)$$

Then it cannot be the case that $S_N(b) = b$; for if this were so, then under N, an agent of type a could report b and obtain:

$$v_a(o_b) + T_N(b) - ab = v_a(o_b) + T_H(b) - ab > U_N(a)$$

which contradicts the definition of $U_N(a)$ as the highest utility a could achieve under N.

Hence, $S_N(b) = c$ for some $c \neq b$, and

$$T_N(b) - bc = T_H(b)$$

Now consider the utility that a would receive from reporting c under N. It would be:

$$v_a(o_b) + T_N(c) - ac = v_a(o_b) + T_H(b) + bc - ac \geq$$

$$v_a(o_b) + T_H(b) - ab \geq U_H(a) > U_N(a)$$

and we have our contradiction.

VTFU condition being violated $\implies$ RP does not hold with fixed transfers and variable utilities.. Choose an instance where the VTFU condition is violated, i.e., there exist $a, b, c \in \Theta$ such that:

$$ac > ab + bc$$

Choose an arbitrary outcome o and let $F(\cdot) = o$. Let $\epsilon = (ac - ab - bc)/2 > 0$. N defined by:

$$A_N(\cdot) = o$$

$$T_N(c) = ab + bc + \epsilon$$

$$T_N(\theta) = 0 \text{ for } \theta \neq c$$

implements F and is non-truthful as b will report c to obtain a utility of $v_b(o) + ab + \epsilon$. On the other hand a will report truthfully and thus obtain a utility of $v_a(o)$ (since $ac > ab + bc + \epsilon$).

Consider any mechanism H with $A_H(\cdot) = o$ and for which each type θ, if it reports truthfully, gets utility $U_N(\theta)$ (so that $T_H(b) = ab + \epsilon$ and $T_H(\theta) = 0$ for $\theta \neq b$). If the revelation principle is to hold, this mechanism must be truthful. But in fact, by reporting b, a can obtain a utility of $v_a(o) + \epsilon > v_a(o) = U_N(a)$. So the revelation principle does not hold. □

4.3.1. Revisiting the Running Example. The mechanism in 4.2.2 maximized the city's objective but resulted in uneven utilities for the different types of agent: *North* and *East* each obtained 7 while *West* only obtained 3 and *South* only 1. Suppose now the city would like equal utilities for all types. (Note that, because the city can make arbitrary transfers, any mechanism where all types receive utility u_0 can easily be transformed into another mechanism where all types receive utility u_1 and that implements the same choice function, simply by adding $u_1 - u_0$ to each transfer.) The best truthful mechanism is the following, resulting in an objective of 30 while ensuring each type achieves a utility of exactly 3 (again, this utility can easily be transformed into any other number).

$H =$

	$\hat{North}$	$\hat{West}$	$\hat{East}$	$\hat{South}$
A	*fiber*	*vitamins*	*protein*	*protein*
T	2	0	0	2

But might there be a non-truthful mechanism that achieves a higher objective? When we check the VTFU condition, we see that the following triples violate the triangle inequality: (*North, West, East*) and (*North, West, South*). Thus, the revelation principle does not hold. Indeed, in this case there is in fact a better non-truthful mechanism. Consider the following non-truthful mechanism under which *West* travels to the *South* food bank. It results in an objective of 35 while still giving all types utility 3.

$N =$

	$\hat{North}$	$\hat{West}$	$\hat{East}$	$\hat{South}$
A	*fiber*	*fiber*	*protein*	*protein*
T	2	0	0	2

4.4. Fixed Transfers, Variable Utilities

Definition 4.6 (*FTVU condition*). An instance satisfies the *FTVU condition* if for all ordered triples a, b, c of types,

$$bc < \infty \implies ac \leq ab$$

THEOREM 4.7. *The RP holds with fixed transfers and variable utilities iff the FTVU condition holds.*

PROOF.

FTVU condition $\implies$ *RP holds with fixed transfers and variable utilities.* By way of contradiction, assume there exists an instance where the FTVU condition holds, but the RP does not. That is, there exists a mechanism N, together with a strategy S_N, such that the mechanism H where $A_H = A_N \circ S_N \triangleq F$ and $T_H = T_N \circ S_N$ is not truthful. That is, using the fact that misreporting always comes at a nonnegative cost, there must exist types a, b with $F(a) = o_a \neq o_b = F(b)$ such that a prefers misreporting to b, that is:

$$v_a(o_b) + T_H(b) - ab > v_a(o_a) + T_H(a)$$

Moreover, since N together with S_N does implement F, there must exist some $c \in \Theta$ such that $S_N(b) = c$, $A_N(c) = o_b$, and $S_N(a) \neq c$. Now, by the FTVU condition, for any c that is reportable by b, $ac \leq ab$. Hence,

$$v_a(A_N(c)) + T_N(c) - ac = v_a(o_b) + T_H(b) - ac$$

$$\geq v_a(o_b) + T_H(b) - ab > v_a(o_a) + T_H(a) \geq$$

$$v_a(A_N(S_N(a))) + T_N(S_N(a)) - c(a, S_N(a))$$

Hence, under N, by reporting c, a can achieve a higher utility than by following S_N, contradicting the optimality of s.

FTVU condition being violated $\implies$ RP does not hold with fixed transfers and variable utilities. Let $a, b, c \in \Theta$ violate the FTVU condition. That is, $bc < \infty$ and $ac > ab$. We will show that we can choose an F and N such that N (with associated optimal strategy S_N) implements F, but H where $A_H = F$ and $T_H = T_N \circ S_N$ is not truthful. In fact we will show that there is even a *constant* function F for which this is the case, which we will use in a later proof. (Other than this, the second part of the proof of Theorem 4.8 will also prove the claim being proven here.)

Choose an arbitrary outcome o and let $F(\cdot) = o$. Let $\epsilon = ac - ab$. The N defined as follows trivially non-truthfully implements F.

$$A_N(\cdot) = o$$

$$T_N(c) = bc + (ab + ac)/2$$

$$T_N(a) = bc$$

$$T_N(\cdot) = 0 \text{ everywhere else}$$

Note $S_N(a) = a$ because $(ab + ac)/2 < ac$, and $S_N(b) = c$ because $bc + (ab + ac)/2 > bc$.

Now consider H where $A_H = o$ and $T_H = T_N \circ S_N$. We have:

$$T_H(b) = bc + (ab + ac)/2$$

$$T_H(a) = bc$$

But then, a can improve its utility by $(ab + ac)/2 - ab > 0$ by misreporting b. Thus H does not truthfully implement F. $\square$

4.4.1. Special Case: Zero Transfers. We now consider the special case where we have no transfers at all; that is, transfers are fixed at zero. Of course, if the FTVU condition holds in general, then the revelation principle will also hold for this special case. However, we may wonder whether the full FTVU condition is necessary for the revelation principle to hold in this special case, or if a more relaxed condition will do for the case of zero transfers. It turns out that the full FTVU condition is still necessary, as we now show. (This will no longer be true if we know the valuation function, as we will show in 5.3.1.)

THEOREM 4.8. *The RP holds with zero transfers and variable utilities iff the* FTVU condition *holds.*

PROOF.

The FTVU condition *holding $\implies$ the RP with zero transfers and variable utilities holds.* This follows immediately from Theorem 4.7 as zero transfers is a special case of fixed transfers.

The FTVU condition *not holding $\implies$ the RP with zero transfers and variable utilities does not hold.* Let $a, b, c \in \Theta$ violate the FTVU condition. That is, $bc < \infty$ and $ac > ab$. We will show that we can define outcomes and valuation functions for the types, as well as a non-truthful mechanism N (with an associated optimal strategy S_N) without transfers that implements a choice function F that no truthful mechanism without transfers would implement. That is, H with $A_H = A_N \circ S_N$ and $T_H = 0$ is not truthful.

Create outcomes o_a and o_b and let $v_a(o_a) = 0$, $v_a(o_b) = (ab + ac)/2$, $v_b(o_a) = 0$, $v_b(o_b) = bc + 1$. Consider the mechanism N defined by $T_N = 0$, $A_N(a) = A_N(b) = o_a$, $A_N(c) = o_b$, and $A_N(\cdot) = o_a$ everywhere else. Associated with it is some optimal strategy S_N for which $S_N(a) = a$ and $S_N(b) = c$. However, a mechanism H with $T_H = 0$, $A_H(a) = o_a$, and $A_H(b) = o_b$ is not truthful, because $v_a(o_b) - ab = (ac - ab)/2 > 0 = v_a(o_a)$. □

4.4.2. Revisiting the Running Example. In this setting, the city no longer cares that the types all receive the same utility, but now the city is unable to make welfare transfers. That is, transfers are fixed at zero. The *FTVU condition* does not hold for this cost function: the following triples all violate the condition: (*North, West, East*), (*North, West, South*), (*West, South, East*), and (*East, South, West*). Thus, we may wonder whether a non-truthful mechanism exists that is better than any other mechanism. However, it turns out that the revelation principle still holds *for the agent's specific valuation function* that we are considering here, and so in fact we can restrict attention to truthful mechanisms.

4.5. Fixed Transfers, Fixed Utilities

Definition 4.9 (*FTFU condition*). An instance satisfies the *FTFU condition* if for all ordered pairs a, b of types,

$$ab \in \{0, \infty\}$$

and for all ordered triples a, b, c of types,

$$(ab = 0 \wedge bc = 0) \implies ac = 0$$

Indeed, in the partial verification case (which, in our notation, is the case where for all $a, b \in \Theta, ab \in \{0, \infty\}$), the case where $(ab = 0 \wedge bc = 0) \implies ac = 0$ is known as the *nested range condition* [Green and Laffont 1986], which is known to characterize when the revelation principle holds in that case, *if* the transfers are considered part of the choice function (i.e., they are fixed); note that in this case utilities are also necessarily fixed because no agent will ever incur nonzero reporting costs.

THEOREM 4.10. *The RP holds with fixed transfers and fixed utilities iff the FTFU condition holds.*

Proof appears in the appendix.

5. REVELATION PRINCIPLE FOR KNOWN VALUATIONS

So far, we have always considered whether the revelation principle holds for a given combination of Θ and $c : \Theta \times \Theta \to \mathbb{R}$. For it to hold meant that *no matter what* the valuation function v and the choice function F (and, possibly, the transfer and/or utilities to be achieved) are, it is either truthfully implementable or not implementable at all. But if we have a particular valuation function in mind, we may not care whether the revelation principle holds for other valuation functions; we just want to know whether *for this valuation function* we can restrict our attention to truthful mechanisms. As we already alluded to in 4.4.2, the revelation principle may hold for a specific valuation function even when it does not hold for all valuation functions. Accordingly, in this section, we consider the case where the valuation function is known (or fixed). Hence, in this section, an *instance* consists of not only Θ and c, but also O and v. Later, in Section 6, we will consider the case where *everything* is fixed, including the choice function (and, possibly, the transfer and/or utilities to be achieved).

5.1. Known Valuations, Variable Transfers, Variable Utilities

Finding useful conditions that ensure that the RP holds in this case is currently an open problem. We conjecture that verifying whether the revelation principle holds in this case is NP-complete.

5.2. Known Valuations, Variable Transfers, Fixed Utilities

It turns out that in the VTFU case, it makes no difference whether we know the valuation function.

THEOREM 5.1. *The RP holds with known valuations, variable transfers, and fixed utilities iff the VTFU condition holds.*

PROOF.

VTFU condition holds $\implies$ *RP holds.* By Theorem 4.5, the VTFU condition implies that the RP holds for all possible valuation functions, so it will continue to hold for a specific valuation function.

VTFU condition being violated $\implies$ *RP does not hold.* In the corresponding part of the proof of Theorem 4.5 (without known valuations), we used a constant choice function whose choice of outcome did not matter. Hence, this part of the proof carries over unmodified to this case. □

5.3. Known Valuations, Fixed Transfers, Variable Utilities

THEOREM 5.2. *The RP holds with known valuations, fixed transfers and variable utilities iff the FTVU condition holds.*

PROOF.

FTVU condition holds $\implies$ *RP holds.* By Theorem 4.7, the FTVU condition implies that the RP holds for all possible valuation functions, so it will continue to hold for a specific valuation function.

FTVU condition being violated $\implies$ *RP does not hold.* In the corresponding part of the proof of Theorem 4.7 (without known valuations), we used a constant choice function whose choice of outcome did not matter. Hence, this part of the proof carries over unmodified to this case. (Note this is not true for the case where transfers must be *zero*—Theorem 4.8—and as we will see next, the condition does change in that case.) □

5.3.1. Special Case: Zero Transfers. We now again consider the special case of zero transfers. Unlike in the unknown-valuations case, here the condition turns out not to be the same as in the general fixed-transfers case.

For example, consider a setting with a single outcome o. Then, even if the cost function violates the FTVU condition, the RP still holds when transfers must be 0, because there is only a single mechanism, in which every type gets the same outcome.

Thus, we provide a separate condition for this case. It is not elegant, but, as we show, it can still be checked in polynomial time.

Definition 5.3 (*KZTVU condition*). An instance satisfies the *KZTVU condition* if for all ordered triples a, b, c of types, there exists no two outcomes o_a and o_b, type $\theta_a \neq c$, and allocation function $A : \Theta \to O$, such that the following hold.

$$o_b \neq o_a \tag{1}$$

$$v_a(o_b) - ab > v_a(o_a) \tag{2}$$

$$A(\theta_a) = o_a \tag{3}$$

$$(\forall \theta)\ v_b(o_b) - bc \geq v_b(A(\theta)) - b\theta \tag{4}$$

$$(\forall \theta)\ v_a(o_a) - a\theta_a \geq v_a(A(\theta)) - a\theta \tag{5}$$

Note that if the FTVU condition holds on a triple a, b, c, then the KZTVU condition also holds for that triple; but the converse is not true as the single-outcome example above illustrates.

THEOREM 5.4. *The RP holds with known valuations, zero transfers and variable utilities iff the* KZTVU condition *holds.*

Proof appears in the appendix.

5.4. Known Valuations, Fixed Transfers, Fixed Utilities

THEOREM 5.5. *The RP holds with known valuations, fixed transfers, and fixed utilities iff the FTFU condition holds.*

Proof appears in the appendix.

6. REVELATION PRINCIPLE FOR FULLY SPECIFIED INSTANCES

In the previous section, we considered the case where we already know the valuation function, and wish to know if the revelation principle holds for that valuation function. Still, all that is needed to violate of the revelation principle is that there is *some* choice function (possibly together with transfers and/or utilities) that can be nontruthfully, but not truthfully, implemented. But this may be of little interest if we already know the precise choice function (etc.) we wish to implement. Indeed, even if the revelation principle does not hold for all choice functions (etc.), it may yet hold for the one we care about. This is what we study in this section. Hence, an instance is now a *fully specified instance*, consisting of Θ, c, O, and v as before, but also F, possibly a specific transfer function $T^* : \Theta \to \mathbb{R}$, and possibly a specific utility function $U^* : \Theta \to \mathbb{R}$, which we wish to implement.

Definition 6.1 (*Revelation Principle on an Fully Specified Instance*). We say the RP is true for a fully specified instance if either (1) there either a truthful mechanism T that implements the choice function (with the required utilities and/or transfers), or (2) no non-truthful mechanism N that does.

As it turns out, deciding whether the RP holds on individual fully specified instances comes down to the computational problem of deciding whether a nontruthful implementation exists. The following lemma makes this clear.

LEMMA 6.2. *Determining whether the revelation principle fails to hold on a given fully specified instance is as hard as determining whether there is a (not necessarily truthful) implementation for that instance.*

PROOF. In each case, we can efficiently verify whether there is a truthful implementation of that instance: if there are no transfers, there is only one mechanism to verify; if there are transfers but they are fixed (or implicitly fixed because utilities are), again there is only one mechanism to verify. Finally, if neither transfers nor utilities are fixed, then it is a simple linear feasibility problem to determine whether transfers exist that implement the choice function.

Hence, we can reduce the problem of determining whether a (not necessarily truthful) implementation exists for an instance to the RP in a fully specified instance problem, as follows. First, check whether a truthful implementation exists; if so the answer is "yes." Otherwise, there is an implementation if and only if the revelation principle fails to hold on this instance. □

THEOREM 6.3. *Computing whether the revelation principle holds on a given fully specified instance is coNP-complete (whether or not transfers and / or utilities are fixed).*

PROOF. The problem of determining whether a (not necessarily truthful) implementation for an instance exists is NP-complete in all these cases [Auletta et al. 2011; Kephart and Conitzer 2015]. Hence, the result follows immediately from Lemma 6.2. □

7. CONCLUSIONS

In this paper, we studied mechanism design with costly reporting. Because the revelation principle is the foundation for so much of existing mechanism design theory, we focused on determining necessary and sufficient conditions for it to hold, under various circumstances (allowing transfers/utilities to vary or requiring them to remain fixed from the nontruthful to the truthful implementation; knowing the valuation function/choice function, or not).

We believe that the framework of mechanism design with costly reporting is one that will be of increasing importance. This is because the parties that run mechanisms increasingly have data on the other agents, as opposed to knowing nothing about them *ex ante* and only being able to ask them about their preferences. Now, an agent can often change the data that the mechanism has about it. We discussed several examples in the introduction; another possibility is for the agent to actively (and possibly selectively) avoid having data collected on it, for example by logging out of systems, erasing cookies, avoiding using credit cards that identify them, filing "right to be forgotten" requests, etc. All of these, though, come at some effort (or other) cost. Hence, the standard mechanism design framework where an agent can report any type at no cost—the "anonymous bidder walking into a Sotheby's auction" model[10]—does not exactly fit such applications. But the mechanism design with reporting framework does.

Acknowledgments

We are thankful for support from ARO under grants W911NF-12-1-0550 and W911NF-11-1-0332, NSF under awards IIS-1527434, IIS-0953756, CCF-1101659, and CCF-1337215, and a Guggenheim Fellowship. This work was done in part while the authors were visiting the Simons Institute for the Theory of Computing.

[10]Of course, the standard mechanism design framework where misreporting is costless can perfectly well address situations where the party running the mechanism has prior information over the agent. The point is that the standard framework does *not* address the agent being able to *change* this prior information at some cost.

REFERENCES

Vincenzo Auletta, Paolo Penna, Giuseppe Persiano, and Carmine Ventre. 2011. Alternatives to truthfulness are hard to recognize. *Autonomous Agents and Multi-Agent Systems* 22, 1 (2011), 200–216.

Marco Barreno, Blaine Nelson, Anthony D. Joseph, and J. D. Tygar. 2010. The security of machine learning. *Machine Learning* 81, 2 (2010), 121–148.

Jesse Bull. 2008. Mechanism Design with Moderate Evidence Cost. *The B.E. Journal of Theoretical Economics* 8, 1 (May 2008), 1–20.

Ioannis Caragiannis, Edith Elkind, Mario Szegedy, and Lan Yu. 2012. Mechanism design: from partial to probabilistic verification. In *Proceedings of the ACM Conference on Electronic Commerce (EC)*. Valencia, Spain, 266–283.

Vincent Conitzer and Tuomas Sandholm. 2002. Complexity of Mechanism Design. In *Proceedings of the 18th Annual Conference on Uncertainty in Artificial Intelligence (UAI)*. Edmonton, Canada, 103–110.

Vincent Conitzer and Tuomas Sandholm. 2004. Self-interested Automated Mechanism Design and Implications for Optimal Combinatorial Auctions. In *Proceedings of the ACM Conference on Electronic Commerce (EC)*. New York, NY, USA, 132–141.

Nilesh N. Dalvi, Pedro Domingos, Mausam, Sumit K. Sanghai, and Deepak Verma. 2004. Adversarial Classification. In *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. Seattle, WA, USA, 99–108.

Raymond Deneckere and Sergei Severinov. 2007. Optimal Screening with Costly Misrepresentation. (2007). Working paper available at http://www.severinov.com/working_papers/screening_costly_misrepresentation_jul07.pdf.

Jerry Green and Jean-Jacques Laffont. 1986. Partially verifiable information and mechanism design. *Review of Economic Studies* 53 (1986), 447–456.

Moritz Hardt, Nimrod Megiddo, Christos Papadimitriou, and Mary Wootters. 2016. Strategic Classification. In *Innovations in Theoretical Computer Science (ITCS)*. Cambridge, MA, USA.

Navin Kartik. 2008. *Strategic Communication with Lying Costs*. 2008 Meeting Papers 350. Society for Economic Dynamics. https://ideas.repec.org/p/red/sed008/350.html

Navin Kartik and Olivier Tercieux. 2012. Implementation with evidence. *Theoretical Economics* 7, 2 (2012), 323–355.

Andrew Kephart and Vincent Conitzer. 2015. Complexity of Mechanism Design with Signaling Costs. In *Proceedings of the Fourteenth International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS)*. Istanbul, Turkey, 357–365.

Jeffrey Lacker and John Weinberg. 1989. Optimal Contracts under Costly State Falsification. *Journal of Political Economy* 97, 6 (1989), 1345–63.

Jean-Charles Rochet. 1987. A necessary and sufficient condition for rationalizability in a quasi-linear context. *Journal of Mathematical Economics* 16, 2 (April 1987), 191–200.

Michael Spence. 1973. Job Market Signaling. *Quarterly Journal of Economics* 87, 3 (1973), 355–374.

Lan Yu. 2011. Mechanism design with partial verification and revelation principle. *Autonomous Agents and Multi-Agent Systems* 22, 1 (2011), 217–223.