Limited Verification of Identities to Induce False-Name-Proofness

Vincent Conitzer
Department of Computer Science
Duke University
Durham, NC 27708, USA
conitzer@cs.duke.edu

Abstract

In open, anonymous environments such as the Internet, mechanism design is complicated by the fact that a single agent can participate in the mechanism under multiple identifiers. One way to address this is to design *false-name-proof* mechanisms, which choose the outcome in such a way that agents have no incentive to use more than one identifier. Unfortunately, there are inherent limitations on what can be achieved with false-name-proof mechanisms, and at least in some cases, these limitations are crippling. An alternative approach is to verify the identities of all agents. This imposes significant overhead and removes any benefits from anonymity.

In this paper, we propose a middle ground. Based on the reported preferences, we check, for various subsets of the reports, whether the reports in the subset were all submitted by different agents. If they were not, then we discard some of them. We characterize when such a limited verification protocol *induces* false-name-proofness for a mechanism, that is, when the combination of the mechanism and the verification protocol gives the agents no incentive to use multiple identifiers. This characterization leads to various optimization problems for minimizing verification effort. We study how to solve these problems. Throughout, we use combinatorial auctions (using the Clarke mechanism) and majority voting as examples.

1 INTRODUCTION

There are many important settings in which a decision must be made on the basis of multiple agents' preferences. For example, in (combinatorial) auctions, we typically want to allocate the item(s) in a way that maximizes the sum of the bidders' valuations.[1] As another example, in elections, we typically want to elect an alternative that many of the voters prefer to many of the other alternatives. A naïve approach to aggregating the agents' preferences is to simply ask each agent to report her preferences, and then choose the optimal outcome for the reported preferences. Unfortunately, under this naïve approach, an agent may have an incentive to lie about her preferences to obtain an outcome that is better for herself. This can be detrimental to our objective, because an outcome that is good with respect to the reported preferences need not be good with respect to the true preferences. *Mechanism design* studies how to choose outcomes so that good results (with respect to the true preferences) are obtained even when agents behave selfishly. By a result known as the *revelation principle* [Gibbard, 1973; Green and Laffont, 1977; Myerson, 1979, 1981], it suffices to consider mechanisms under which 1) agents reveal their preferences directly to the mechanism, and 2) agents have no incentive to misreport their preferences. Within this framework, both positive and negative results have been derived. The positive results consist of mechanisms that have some desirable properties—for example, the *Clarke* mechanism [Clarke, 1971], which chooses the optimal allocation for the reported preferences, is *strategy-proof* (no agent ever has an incentive to misreport) due to payments that the agents must make, and has several other nice properties. (The Clarke mechanism is a generalization of the *Vickrey* auction [Vickrey, 1961], and is itself generalized by the class of *Groves* mechanisms [Groves, 1973]. Due to this, the Clarke mechanism is sometimes also referred to as the Generalized Vickrey Auction (GVA) or the Vickrey-Clarke-Groves (VCG) mechanism.) The negative results show that in some settings, no mechanism exists that has all of a list of properties—for example, the Gibbard-Satterthwaite theorem [Gibbard, 1973; Satterthwaite, 1975] shows that in elections with at least three alternatives and unrestricted preferences, any deterministic strategy-proof mechanism is either *dictatorial* (it always elects the most-preferred alternative of the same voter) or rules out certain

[1]An alternative objective is to maximize expected revenue [Myerson, 1981; Goldberg *et al.*, 2001].

alternatives *ex ante*, that is, *no* votes will make such an alternative win.

While mechanism design has focused mostly on addressing the problem of agents reporting false preferences, there are other ways in which agents may manipulate the mechanism. For example, in open, anonymous environments such as the Internet, it is possible for an agent to participate in the mechanism under multiple identifiers (*e.g.* multiple e-mail addresses) [Yokoo *et al.*, 2001, 2004]. The party running the mechanism (known as the *center*) generally does not know whether multiple identifiers correspond to the same agent, so each identifier must be treated as a separate agent. Because of this, using multiple identifiers can be beneficial even if the mechanism is strategy-proof.

One way to address this is to design mechanisms that are *false-name-proof* [Yokoo *et al.*, 2001, 2004]. A mechanism is false-name-proof if no agent ever has an incentive to use more than one identifier. Various false-name-proof mechanisms have been designed for combinatorial auctions [Yokoo *et al.*, 2001; Yokoo, 2003; Yokoo *et al.*, 2006; Matsuo *et al.*, 2006]. Unfortunately, none of these mechanisms are completely satisfactory, in the following sense: it is known that it is impossible for a false-name-proof mechanism to always choose the efficient (value maximizing) allocation [Yokoo *et al.*, 2004].

For other settings, there seems to be little hope of creating any reasonable false-name-proof mechanism. Consider the simple example of voting over which one of two alternatives, a and b, to elect. If it is not possible to submit false names, then this setting does not pose any problems from a mechanism design perspective: the majority rule (choose the alternative that is preferred by more voters) is strategy-proof. On the other hand, if it is possible to use false names, then each voter can submit an unlimited number of votes for either alternative. Hence, the number of votes submitted for an alternative becomes almost[2] meaningless.

It is clear that false-name-proof mechanisms provide us with only limited options for running mechanisms in open, anonymous environments. Fortunately, in practice, there are often ways to check that preference reports (such as bids or votes) came from different agents. One way is to make (some of) the agents submit verifiable real-world identifiers, such as credit card numbers, phone numbers,[3] *etc.* Care must be taken that one agent cannot use multiple real-world identifiers that she owns: we must check that the credit cards or phone numbers belong to different agents, or at least argue that it would be economically impractical for an agent to own sufficiently many credit cards or phone numbers to have a significant impact on the outcome of the mechanism. Care must also be taken that an agent cannot use the real-world identifiers of unwilling other agents: we can call the phone number to check that its owner is aware of its use in the mechanism, or place an insignificant charge on the credit card. Of course, if another agent is willing to let her real-world identifier be used in this way, then there does not appear to be much that can be done—but this is more akin to collusion than to submitting false names. While collusion is an important problem, we will not address it in this paper.

[2]If *nobody* prefers alternative a, this will still be reflected in a's number of votes (zero). Therefore, one false-name-proof mechanism is the following "unanimity" mechanism: if all voters prefer the same alternative, elect that alternative; otherwise, randomly elect an alternative. Clearly, this is not a very satisfactory voting mechanism. Nevertheless, by a general characterization of false-name-proof voting mechanisms [Conitzer, 2007], it is in fact the best false-name-proof mechanism for this setting.

[3]For example, one way of signing up for a Gmail™ account is to submit a mobile phone number, to which an invitation code will then be sent. This is explicitly done to prevent a single person from signing up for many accounts.

We can also think of other, perhaps more creative ways of checking that reports came from different agents. For example, we can require a subset of the agents to appear in an Internet chat room at a certain time, and to each (simultaneously) carry on a short, insignificant conversation with the center. Presumably, it is extremely difficult to pretend to be two different agents and carry on two different conversations at once: it would be noticeable that whenever one of the two agents is typing, the other is not. (Moreover, artificial intelligence is not yet at the level that a computer could carry on the conversation, *cf.* the Turing Test.) One could get a friend to act as the other agent, but again, this would be more akin to collusion.

If there is a way to check that reports came from different agents, then we can ensure that agents have no incentive to use multiple identifiers, as follows. After the reports have been submitted, check that all of them came from different agents. If not, choose an outcome that is a worst-case scenario for all agents—burn all the items in the auction, elect a far underqualified candidate as president, *etc.* This draconian approach is unsatisfactory for several reasons. The first reason is that using the worst-case outcome may not be feasible in practice. It may be too risky: if even one agent decides to submit false names anyway (for example, because she (erroneously!) believes that she can circumvent the verification mechanism, or because she has not understood the incentives), the worst-case outcome will result. It will also be difficult to commit to using the worst-case outcome: one can imagine that the center will want to backtrack on burning the items. A more satisfactory solution would be to simply discard (some of) the false-name reports and run the mechanism again with the remaining ones, but the effect of this on incentives is less clear.

The second reason is that this approach requires the verification of all submitted reports. This seems excessive—for instance, do we really need to check bids that did not win? It would be much preferable to do only the minimum amount of verification that is necessary to make using multiple identifiers strategically suboptimal, thereby preserving as much anonymity as possible and minimizing verification effort.

In this paper, we propose an approach where only limited subsets of the reports are checked to see whether they came from different agents. Reports that are not confirmed are simply discarded. We characterize when such a limited verification protocol incentivizes agents to use only a single identifier, and study how to minimize verification effort given this characterization.

2 DEFINITIONS

In this section, we cover the required mechanism design background and define verification protocols.

2.1 MECHANISM DESIGN

A mechanism design setting can be described as follows. There is a set of *agents*, and a set of *outcomes* that we must decide among. Each outcome specifies information such as the elected alternative, the allocation of items, the payment to be made by each agent, *etc.* In settings where the set of agents is initially unknown to the center, the outcome space is often, to some extent, also unknown (*e.g.* we do not know which agents will be around to make payments or win items). Because of this, in a slight deviation from standard notation, let O denote the set of outcomes *for a single agent* participating in the mechanism. For example, in a combinatorial auction (which will be defined shortly), an element of O is a subset of the items (that the agent will receive) and a payment (to be made by the agent). (This is assuming *no externalities*, that is, an agent does not care about what the other agents receive or pay.) On the other hand, if the agents are voting over multiple alternatives, an element of O specifies which alternative is elected. Since we will be concerned with anonymous settings, O should be the same for each agent.

Each agent has privately held preferences over O. This is formalized as follows: each agent has a (privately held) *type* $\theta \in \Theta$, which encodes her preferences, and there is a (commonly known) *utility function* $u : \Theta \times O \rightarrow \mathbb{R}$ relating these types to utilities. That is, $u(\theta, o)$ is the utility for an agent of type θ for receiving outcome o.

A *direct revelation mechanism* is defined as follows. Each agent submits a report $r \in \Theta$ of her type, after which a function f (the mechanism) chooses an outcome for each agent based on these reports. That is, if R is the (multi)set of all submitted reports, then an agent that reported $r \in R$ receives outcome $f(r, R - \{r\}) \in O$. (Again, this is a slight deviation from standard notation.) We note that not all such functions f define a sensible mechanism: for example, some functions f would award the same item to multiple agents, or elect different alternatives for different agents. However, all that is important for our purposes is that every sensible mechanism can be described in this way. Randomized mechanisms can be accommodated by making O the set of all distributions over (deterministic) outcomes for a single agent.

A mechanism is *(ex-post) individually rational (IR)* if participating in the mechanism is never harmful to an agent (assuming she reports her true type). That is, for any type θ and any (multi)set S of reports by other agents, $u(\theta, f(\theta, S)) \geq u(\theta, f(\emptyset, S))$. A mechanism is *strategy-proof* if no agent ever benefits from misreporting her type. That is, for any $\theta, \theta' \in \Theta$, for any S, $u(\theta, f(\theta, S)) \geq u(\theta, f(\theta', S))$. Throughout, we will restrict our attention to mechanisms f that are both IR and strategy-proof, so that we need not worry about an agent submitting no reports at all or submitting a single false report.

2.1.1 Example: combinatorial auctions using the Clarke mechanism

In a *combinatorial auction*, there is a set of items I simultaneously for sale. Assuming no externalities, an outcome for an agent (bidder) in a combinatorial auction consists of a subset of the items, plus a payment that she has to make. That is, $O = 2^I \times \mathbb{R}$. We will assume that bidders have *quasilinear* preferences, that is, $u(\theta, (I', \pi)) = v(\theta, I') - \pi$. We will be especially interested in *single-minded* bidders, for whom there is some $I'' \subseteq I$ such that $v(\theta, I') = v(\theta, I'')$ if $I'' \subseteq I'$, and $v(\theta, I') = 0$ otherwise.

Of course, the mechanism cannot allocate the same item to multiple bidders. Typical mechanisms allocate the items to maximize *efficiency*, that is, the sum of the bidders' valuations, $\sum_i v_i(\theta_i, I_i)$ (where I_i is the set of items allocated to i).[4] Items may remain unallocated (*free disposal* is allowed). The problem of determining who wins what is (in general) NP-hard [Rothkopf *et al.*, 1998], even to approximate [Håstad, 1999; Sandholm, 2002], though in practice it can typically be solved reasonably fast [Sandholm *et al.*, 2006; Sandholm, 2006].

We still need to specify the bidders' payments. The *Clarke* mechanism [Clarke, 1971] determines the payments as follows: bidder i pays $\sum_{j \neq i} v_j(\theta_j, I_j^{-i}) - v_j(\theta_j, I_j)$, where I_j^{-i} is the set of items that would have been allocated to j if i had not been present. That is, each bidder pays the amount by which her presence makes the other bidders worse off (in terms of the allocation of items). Combinatorial auctions that use the Clarke mechanism (also known as Generalized Vickrey Auctions) are well-known to be IR and strategy-proof.

[4]Mechanisms that maximize expected revenue do not always maximize efficiency, but we will not concern ourselves with revenue in this paper.

2.1.2 Example: majority voting with two alternatives

Suppose we need to decide between two alternatives, a and b. The agents (voters) can *vote* over these alternatives, declaring a preference either for a or for b. We can use the *majority rule*: the alternative that receives more votes wins (if there is a tie, we choose the winner randomly). The majority rule is IR and strategy-proof. (By the Gibbard-Satterthwaite impossibility result [Gibbard, 1973; Satterthwaite, 1975] mentioned above, under minor restrictions on the mechanism, strategy-proofness cannot be obtained when there are three or more alternatives, unless there are restrictions on the preferences of the voters.)

2.2 FALSE-NAME-PROOFNESS

In open, anonymous environments such as the Internet, we are confronted with the issue that a single agent may use multiple identifiers to submit multiple reports (and the center will not know that they came from the same agent). In combinatorial auctions using the Clarke mechanism, such *false-name bidding* is known to sometimes be advantageous [Yokoo *et al.*, 2004]. For example, if one bidder bids $(\{a, b\}, 1)$ (that is, 1 on the bundle of items $\{a, b\}$), then a second bidder can obtain both items by bidding $(\{a, b\}, k)$ with $k > 1$ to obtain both items at a price of 1, but she would be better off submitting two bids $(\{a\}, 2)$ and $(\{b\}, 2)$, in which case she would obtain both items and pay 0. Moreover, this phenomenon occurs not just under the Clarke mechanism, but under *any* mechanism that allocates the items efficiently [Yokoo *et al.*, 2004]. Also, in majority voting, submitting multiple votes for one's preferred alternative can obviously be beneficial.

We denote the outcome that an agent obtains by submitting a set of reports S when the others' reports are $R - S$ by $f(S, R - S)$. A mechanism f that is IR and strategy-proof is *false-name-proof* if for any θ, S, and S', $u(\theta, f(\theta, S')) \geq u(\theta, f(S, S'))$. As we just showed, neither combinatorial auctions using the Clarke mechanism nor majority elections are false-name-proof.

2.3 VERIFICATION PROTOCOLS

We now move on to the contributions of this paper. A *verification protocol* P for a mechanism f works as follows. For every set of reports R, P can *check* any subset $S \subseteq R$. When S is checked, a message is sent to the agents that reported S that they must *confirm* these reports (by submitting a real-world identifier, appearing in a chat room, *etc.*). In response to such a request, each agent can confirm at most one of her reports in S. Thus, if one agent submitted multiple reports in S, the protocol P will discover that someone submitted multiple reports, because some of the reports will not be confirmed. However, the agent that submitted multiple reports can still confirm one of her reports, and there is no way for P to know that this one confirmed report belongs to the agent that submitted multiple reports. P will know, however, that each confirmed report in S came from a different agent.

It sometimes makes sense to check multiple subsets of the reports in sequence. For example, if the center uses the chat room verification protocol described above, perhaps the center can carry on a conversation with at most two agents at the same time (*e.g.* because the center has only two employees available for chatting with agents). In this case, if we first check reports r_1 and r_2, and then reports r_2 and r_3, we still cannot be sure that r_1 and r_3 were not submitted by the same agent. This scenario does not make sense, however, when we verify using real-world identifiers such as credit card or phone numbers (as described above): with this technique, for any two reports for which we have obtained real-world identifiers, we will know whether they were submitted by the same agent. Thus, with this technique, there is effectively only a single subset of reports that is checked.

A protocol is thus defined by a function that takes as input a set of reports R, and as output produces a finite sequence of subsets $S_1, S_2, \ldots, S_k$ of R that it will check, in that order. (We will only consider deterministic verification protocols in this paper.) If for some check S_i, some reports $F \subseteq S_i$ are not confirmed, then the protocol discards F and restarts with $R - F$ as the set of reports. Since the set of reports is finite, the protocol can restart only finitely many times, and thus must eventually terminate. When it does, the outcome that the mechanism f prescribes for the remaining set of reports R^{final} is chosen.

Example 1 *Consider a combinatorial auction with three items* $\{a, b, c\}$ *for sale. In this example, all bidders are* single-minded. *Bidder* 1 *submits a bid of* 4 *for* $\{a, b\}$. *Bidder* 2 *submits a bid of* 4 *for* $\{b, c\}$. *Bidder* 3 *submits a bid of* 5 *for* $\{a\}$. *Bidder* 4 *submits two bids (under different identities): one of* 5 *for* $\{b\}$, *and one of* 5 *for* $\{c\}$. *If we run the Clarke mechanism (without verification), then bidder* 3*'s bid and bidder* 4*'s two bids are accepted, and neither of* 3 *and* 4 *will pay anything (since the mechanism must assume that all bids came from different bidders). Also, if* 4 *had not been able to use multiple identifiers, then* 4 *would have had to pay* 4 *to win* $\{b, c\}$. *With a verification protocol P, the following sequence of events may occur. P checks the set of bids* $\{(\{a\}, 5), (\{b\}, 5)\}$. *Bidders* 3 *and* 4 *confirm these bids. Then, P checks the set of bids* $\{(\{b\}, 5), (\{c\}, 5)\}$. *Because both of these bids were submitted by* 4, *she can confirm only one of them; say she confirms* $(\{b\}, 5)$. *P discards* $(\{c\}, 5)$, *and restarts with the remaining bids.*

The use of a verification protocol changes the incentives for the agents participating in the mechanism. For example, whereas without verification, it may have been beneficial for a bidder to place multiple bids in a combinatorial auction (using multiple identifiers), when the verification protocol is added this may no longer be beneficial, because the bidder may not be able to prevent some of them from being discarded. We now formalize when a verification protocol is effective in discouraging the use of multiple identifiers.

Definition 1 *Let f be an IR and strategy-proof mechanism. We say that verification protocol P* induces *false-name-proofness for f if under the combination (f, P), it is always optimal for an agent to use only a single identifier, report her true type, and always confirm her report when it is checked, given that the others do so as well. That is, behaving honestly is an* ex-post *equilibrium.*

In the next section, we characterize when a verification protocol induces false-name-proofness.

3 CHARACTERIZING VERIFICATION PROTOCOLS THAT INDUCE FALSE-NAME-PROOFNESS

We start with the following definition.

Definition 2 *Given an IR and strategy-proof mechanism f and a (multi)set of reports R, we say that $S \subseteq R$* requires verification *if there exists some $\theta \in \Theta$ such that $u(\theta, f(S, R - S)) > u(\theta, f(\theta, R - S))$.*

Informally, a subset S of the reports R requires verification if an agent could have a type θ such that, if the agent knew that the other agents' reports were $R - S$, then the agent would be better off reporting S than θ.[5] A set S consisting of only a single type θ' never requires verification, because by the strategy-proofness of f, $u(\theta, f(S, R - S)) = u(\theta, f(\theta', R - S)) \leq u(\theta, f(\theta, R - S))$ for all $\theta \in \Theta$. Also, the empty set never requires verification, because by IR, $u(\theta, f(\emptyset, S)) \leq u(\theta, f(\theta, S))$ for all S and $\theta \in \Theta$.

Example 2 *Consider again Example 1. The set of bids $S = \{(\{b\}, 5), (\{c\}, 5)\}$ requires verification, for the following reason. Consider a single-minded type θ that indicates having a utility of 5 for the bundle $\{b, c\}$. $u(\theta, f(S, R - S)) = 5$, because no payment is required of the bids in S. However, $u(\theta, f(\theta, R - S)) = 5 - 4 = 1$, because the bid θ will win $\{b, c\}$ but be required to pay 4. By symmetry, the set of bids $\{(\{a\}, 5), (\{b\}, 5)\}$ also requires verification. Finally, the set of bids $\{(\{a\}, 5), (\{b\}, 5), (\{c\}, 5)\}$ requires verification.*

[5]For combinatorial auctions, Matsuo *et al.* [2006] also consider a notion of which bids are "suspicious" in the sense that they may have been submitted by a false-name bidder. Based on this notion, they design a false-name-proof mechanism (without any verification of identities). Under this mechanism, the items that the Clarke mechanism would have allocated to suspicious bidders are sold to these bidders as a single bundle, by means of a second-price auction with a reserve price.

Example 3 *Consider an election between two alternatives, a and b, using the majority rule. Suppose that there are 4 votes for a and 2 votes for b. Any subset of at least 3 votes for a requires verification, because if these 3 or more votes were submitted by the same voter (who prefers a), that voter is better off having submitted these votes than she would have been submitting only a single vote for a (which would have resulted in a tie at best). Also, any subset of 4 votes for a and 1 vote for b requires verification. By contrast, no subset of 2 votes for a requires verification, because if these 2 votes had been submitted by the same voter (who prefers a), that voter would have been equally well off casting only a single vote for a.*

The goal of the verification protocol will be to verify, for every subset that requires verification, that that subset was not submitted by a single agent. This does *not* require checking that every report in the subset was submitted by a different agent. Rather, it only requires checking that there are at least two reports in the subset that were submitted by different agents. The next definition makes this precise.

Definition 3 *We say that a verification protocol P for an IR and strategy-proof mechanism f* has sufficiently verified *reports R if, for every $S \subseteq R$ that requires verification, there are two reports $s_1, s_2 \in S$ for which the protocol has verified that s_1 and s_2 were submitted by different agents. That is, for every $S \subseteq R$ that requires verification, P has checked a subset $S' \subseteq R$ with $|S \cap S'| \geq 2$ (and all the reports were confirmed).*

We say that a verification protocol is *sufficient* for f if it sufficiently verifies any reports. More precisely:

Definition 4 *We say that a verification protocol P is* sufficient *for an IR and strategy-proof mechanism f if, for any set of reports R, it will check a sequence of subsets that will allow it to sufficiently verify R. That is, for any set of reports R, for any subset $S \subseteq R$ that requires verification, P will eventually check a subset $S' \subseteq R$ with $|S \cap S'| \geq 2$ (unless some report fails to be confirmed before that).*

In the remainder of this section, we show that this notion of sufficiency characterizes when a protocol induces false-name-proofness.

Lemma 1 *If a verification protocol P is sufficient for an IR and strategy-proof mechanism f, then in the set of remaining reports R^{final} that is eventually used to choose the outcome under (f, P), if $S \subseteq R^{final}$ requires verification, then the reports in S cannot all have been submitted by the same agent.*

Proof: Because S requires verification in R^{final}, and P is sufficient for f, P must have checked a subset $S' \subseteq R^{final}$ with $|S \cap S'| \geq 2$. Since no reports in S' failed to be confirmed (as this would have caused them to be discarded), all reports in $S \cap S'$ must have been submitted by different agents. ■

Theorem 1 *If a verification protocol P is sufficient for an IR and strategy-proof mechanism f, then it induces false-name-proofness for f.*

Proof: We must show that it is an *ex-post* equilibrium for each agent to behave honestly, that is, to submit only a single report, consisting of the agent's true type, and to confirm that report whenever the protocol checks it. Let us suppose that all agents other than i behave in this way. Regardless of how agent i acts, when the verification protocol terminates, by Lemma 1, the set of reports submitted by i that were not discarded (call this set S_i^{final}) cannot constitute a set that requires verification. That is, for any true type θ that i may have, $u(\theta, f(S_i^{final}, R^{final} - S_i^{final})) \leq u(\theta, f(\theta, R^{final} - S_i^{final}))$. Moreover, since the other agents ($-i$) behaved honestly, $R^{final} - S_i^{final}$ consists of exactly the reports that the other agents submitted initially. Hence, $u(\theta, f(\theta, R^{final} - S_i^{final}))$ is what agent i would have obtained by behaving honestly. Thus, agent i would have been at least as well off behaving honestly. ■

The converse is also true:

Theorem 2 *If a verification protocol P induces false-name-proofness for an IR and strategy-proof mechanism f, then it is sufficient for f.*

Proof: Suppose that P is not sufficient for f. Then, there must exist a set of reports R and a subset $S \subseteq R$ that requires verification, such that P never checks a subset $S' \subseteq R$ with $|S \cap S'| \geq 2$. Let us consider the situation where each report in $R - S$ is (truthfully) submitted by a different agent (and these agents will always confirm their report when it is checked), and there is one additional agent i that is deciding what to report. Since S requires verification, there exists some $\theta \in \Theta$ that i could have such that $u(\theta, f(S, R - S)) > u(\theta, f(\theta, R - S))$. Moreover, if the agent submits the set of reports S, she can in fact obtain outcome $f(S, R - S)$. This is because P will never (simultaneously) check a pair of reports in S, so the agent can always confirm every report in S (and all of the other agents' reports will always be confirmed as well). Thus, if agent i has type θ, she would prefer submitting the set of reports S over submitting θ. Thus behaving honestly is not an *ex-post* equilibrium, that is, P does not induce false-name-proofness. ■

4 DECIDING WHICH SETS TO CHECK

Now that we know when a protocol induces false-name-proofness, we next investigate how this can be achieved most efficiently—that is, how do we minimize the amount of checking that we do? This question leads to several computational optimization problems that we will study in this section.

4.1 DEFINING THE COMPUTATIONAL PROBLEM

Theorems 1 and 2 show that a verification protocol P for an IR and strategy-proof mechanism f induces false-name-proofness if and only if for every subset of the reports that requires verification, P verifies that at least two of these reports came from different agents (by checking another subset that has at least two reports in common with the former subset). Typically, there are many ways to achieve this. We can take a pair of reports from each subset that requires verification, and check each of these pairs separately. Alternatively, we can check a single subset of the reports that has at least two reports in common with every subset that requires verification. This may not be a decision that we can make: as explained above, in a chat room we may only be able to check subsets of limited sizes, whereas when we use real-world identifiers for verification we are effectively checking only a single subset. Hence, we will study both techniques separately. Still, there is the choice to be made of exactly which subset(s) to check. Presumably, we want to minimize the number of checks, as well as the number of reports in each check, to minimize verification effort and maximize the anonymity of the agents.

One helpful observation is that we can expect reports to always be confirmed, since, after all, the verification protocol induces false-name-proofness. Thus, there is no uncertainty as to what will happen during verification. This also means that we do not care about the order in which we check subsets, rather we only care about which subsets we check.[6]

We can now formalize the question of which subsets to check as a computational problem in (at least) the following ways. In the first problem, we try to minimize the number of subsets checked when each subset checked can have size at most k.

Definition 5 (BOUNDED-SUBSET-VERIFICATION) *We are given a set R ($|R| = m$), a collection of subsets $\mathcal{S} = \{S_1, \ldots, S_n\}$ of R that require verification, a number*

[6]When the verification protocol operates by asking the agents for real-world identifiers, one may have wondered whether it would be more efficient to ask the agents for their real-world identifiers sequentially rather than simultaneously, so that if a report is not confirmed the protocol can restart immediately without first asking the remaining agents for their real-world identifiers. One may also have wondered whether, when the protocol restarts, it would be more efficient to re-use some of the information obtained before the restart. We can now see that both of these ideas would not have contributed to verification efficiency, since we can assume that all reports are always confirmed.

k, and a number t. We are asked whether there exists a collection $\{S'_1, \ldots, S'_t\}$ of subsets of R, such that $|S'_j| \leq k$ for every $1 \leq j \leq t$, and for every $1 \leq i \leq n$, there exists $1 \leq j \leq t$ such that $|S_i \cap S'_j| \geq 2$.

An interesting special case of this problem occurs when $k = 2$: in this case, every S_i must be a superset of some S'_j. The next problem considers the case where we check only one subset, and asks for the smallest subset to check.

Definition 6 (SINGLE-SUBSET-VERIFICATION) *We are given a set R ($|R| = m$), a collection of subsets $\mathcal{S} = \{S_1, \ldots, S_n\}$ of R that require verification, and a number t. We are asked whether there exists a subset S' of R, $|S'| \leq t$, such that for every $1 \leq i \leq n$, $|S_i \cap S'| \geq 2$.*

Without loss of generality, we can assume that for any $i \neq j$, $S_i \not\subseteq S_j$ (if one subset that requires verification is a subset of another subset that requires verification, we can restrict our attention to the former one—that is, we only need to consider the *minimal* subsets that require verification). Also, without loss of generality, we can assume that for any i, $|S_i| \geq 2$ (otherwise the answer would trivially be "no"). Still, one may wonder if we are perhaps considering computational problems that are too general for our specific purpose. Do instances of these computational problems that are derived from (say) bids in a combinatorial auction using the Clarke mechanism have some additional structure that makes them easier to solve? Or is it the case that for *any* instance of the computational problems, that is, any collection $\mathcal{U} = \{U_1, \ldots, U_n\}$ of subsets of a finite set T (where no subset in the collection is a subset of another subset in the collection, and every subset has size at least 2), we can construct a set of bids such that the collection $\mathcal{S}$ of (minimal) subsets of bids that require verification corresponds exactly to $\mathcal{U}$? The next proposition shows that unfortunately, the latter is the case. Thus, at least for combinatorial auctions using the Clarke mechanism, we must consider the above two computational problems in their full generality. To prove the proposition, we first prove some facts about subsets that require verification in combinatorial auctions using the Clarke mechanism. (These facts will also be useful later.) The first lemma shows that if a subset of the bids requires verification, then a single-minded bidder that is tremendously interested in the bundle of items won by that subset of bids would have preferred to submit that subset of bids.

Lemma 2 *Consider a combinatorial auction using the Clarke mechanism. Suppose that $S \subseteq R$ is a subset of the bids that requires verification. Let $I(S)$ be the set of items that S wins. For sufficiently large M, consider a single-minded type θ that values the bundle $I(S)$ (and any supersets of $I(S)$) at M, and everything else at 0. Then $u(\theta, f(S, R - S)) > u(\theta, f(\theta, R - S))$.*

Proof: Because S requires verification, we know that there exists some type $\theta' \in \Theta$ such that $u(\theta', f(S, R - S)) > u(\theta', f(\theta', R - S))$. Now, under the Clarke mechanism, when a bidder faces opponent bids $R - S$, there will be some price $p_{I(S)}(R - S)$ at which she can obtain $I(S)$ (without using false names). Thus, it must be the case that by submitting bids S instead, the sum of the payments charged to these bids is less than $p_{I(S)}(R - S)$ (otherwise there would be no incentive to submit false names for a bidder of type θ'). Now, given that M is sufficiently large, a bidder submitting θ when the other bids are $R - S$ would win the bundle $I(S)$ and pay $p_{I(S)}(R - S)$. But then, a bidder of type θ would prefer submitting bids S and winning $I(S)$ at a lower price. That is, $u(\theta, f(S, R - S)) > u(\theta, f(\theta, R - S))$. ■

The next lemma shows that in combinatorial auctions using the Clarke mechanism, *minimal* subsets that require verification never include losing bids.

Lemma 3 *Consider a combinatorial auction using the Clarke mechanism. Suppose that $S \subseteq R$ is a subset of the bids that requires verification, and $r \in S$ is a losing bid (r receives no items). Then $S - \{r\}$ requires verification.*

Proof: Let $I(S)$ denote the set of items won by the set of bids S. Because S requires verification, by Lemma 2, for sufficiently large M, a single-minded type θ that values the bundle $I(S)$ (and any supersets of $I(S)$) at M, and everything else at 0, will have the property that $u(\theta, f(S, R - S)) > u(\theta, f(\theta, R - S))$.

We also know that $u(\theta, f(S - \{r\}, R - (S - \{r\}))) = u(\theta, f(S, R - S))$, since r receives no items and makes no payment. Now let us consider $f(\theta, R - (S - \{r\}))$—that is, what does the Clarke mechanism do if the bids are $\{\theta\} \cup R - (S - \{r\})$? Because θ places such a high value on $I(S)$ (and on nothing else), θ will win that bundle. The remaining allocation problem is to allocate the remaining items $I - I(S)$ to the bids $R - (S - \{r\})$. The same subproblem occurs when the bids are R, because there the bids in $S - \{r\}$ are awarded $I(S)$, and we are thus left to award $I - I(S)$ to the bids $R - (S - \{r\})$. Thus, the optimal solutions to the remaining allocation problem must be the same in both cases, implying that r does not win when the bids are $\{\theta\} \cup R - (S - \{r\})$. Hence the allocation when the bids are $\{\theta\} \cup R - (S - \{r\})$ is the same as when the bids are $\{\theta\} \cup R - S$.

To determine θ's Clarke payment when the bids are $\{\theta\} \cup R - (S - \{r\})$, we still need to determine the value of the optimal allocation when θ is removed and $R - (S - \{r\})$ remains. Naturally, this value is at least the value of the optimal allocation for bids $R - S$. It follows that θ's Clarke payment when the other bids are $R - S$ is at most θ's Clarke payment when the other bids are $R - (S - \{r\})$. Hence $u(\theta, f(\theta, R - S)) \geq u(\theta, f(\theta, R - (S - \{r\})))$. Now we have established that $u(\theta, f(S - \{r\}, R - (S - \{r\}))) =$

$u(\theta, f(S, R-S)) > u(\theta, f(\theta, R-S)) \geq u(\theta, f(\theta, R - (S - \{r\})))$, and hence $S - \{r\}$ requires verification. ■

We are now ready to prove the proposition.

Proposition 1 *Consider an arbitrary set T ($|T| = m$) and an arbitrary collection of subsets $\mathcal{U} = \{U_1, \ldots, U_n\}$ of T, with the properties that for any $i \neq j$, $U_i \not\subseteq U_j$, and for any i, $|U_i| \geq 2$. There exists a set of single-minded bids in a combinatorial auction such that, when f is the Clarke mechanism, $\mathcal{U}$ corresponds exactly to the collection of minimal subsets of bids that require verification.*

Proof: Let T be the set of items. For every $U_i \in \mathcal{U}$, let there be a bid of value 1 on U_i. In addition, for every $t \in T$, let there be a singleton bid of value 2 on $\{t\}$. Clearly, all the singleton bids win. Moreover, if we remove a singleton bid, then the optimal solution is to accept all the remaining singleton bids (and nothing else). Therefore, under the Clarke mechanism, the singleton bids are charged 0.

Now let us consider which subsets of bids are the minimal subsets that require verification. Because none of the U_i bids win, these bids cannot be part of any minimal subset requiring verification, by Lemma 3. Thus, we can consider subsets S consisting of singleton bids only; let $I(S)$ denote the items that the bids in S bid on. First consider any set S with for all i, $U_i \not\subseteq I(S)$. Then, if a single bidder cast the bids S, that bidder would have been equally well off placing a single bid of value 1 on $I(S)$ instead: this bid would have won (along with the remaining singleton bids), and when this bid is removed, it remains optimal to accept only the remaining singleton bids, for the following reason. Each U_i must intersect with at least one remaining singleton bid because no U_i is contained in $I(S)$, and the singleton bid has a higher value. Thus, the bidder would have had to pay nothing. Hence S does not require verification. Now consider a set S so that $I(S) = U_i$ for some U_i. If a single bidder that very much wanted to obtain U_i cast the bids S, that bidder would have been worse off placing only a single (winning) bid on U_i: this is because when this single bid is removed, the other bid on U_i (of value 1) can be accepted, so that the Clarke payment for the bidder is 1. It follows that any set S of singleton bids so that $I(S) = U_i$ for some U_i requires verification (and that these are the minimal sets that require verification). Because each singleton bid corresponds to a single item, $\mathcal{U}$ corresponds exactly to the collection of minimal subsets that require verification. ■

While the previous proposition shows that combinatorial auctions using the Clarke mechanism can produce any instance of the BOUNDED-SUBSET-VERIFICATION and SINGLE-SUBSET-VERIFICATION problems, this is not necessarily the case for other mechanisms. Consider, for example, voting over two alternatives using the majority rule (as in Example 3).

Proposition 2 *Consider an election between two alternatives, a and b, using the majority rule. If $w \in \{a, b\}$ wins the election by l votes, then the collection $\mathcal{S}$ of minimal subsets that require verification consists of all subsets of $l + 1$ votes for w. If the election is tied, then the collection $\mathcal{S}$ of minimal subsets that require verification consists of all subsets of 2 votes for a, and all subsets that consist of 2 votes for b.*

Proof: For the case where w wins the election, consider any subset of the votes that contains at most l votes for w. If all of these votes were submitted by a single voter that prefers w, then if that voter had submitted only a single vote for w instead, w still would have won (w's score would have decreased by fewer than l votes). If they were all submitted by a voter that prefers the other alternative, obviously they did not make that voter better off (since w won). On the other hand, a subset consisting of $l + 1$ votes for w does require verification, because if all of these votes were submitted by a single voter that prefers w, then if that voter had submitted only a single vote for w instead, w's score would have decreased by l votes, leaving the election tied.

For the case where the election is tied, no subset consisting of at most one vote for each alternative requires verification (it is easy to see that casting such a set of votes cannot make a voter better off than simply casting a vote for her more preferred alternative). On the other hand, a subset consisting of 2 votes for the same alternative (without loss of generality, a) requires verification, because if both of these votes were submitted by a single voter that prefers a, then if that voter had submitted only a single vote for a instead, a's score would have decreased by 1 vote, and b would have won. ■

Knowing a characterization such as the one in Proposition 2 can make solving the computational problems much easier: for example, we can see that for majority voting, if w wins the election by l votes, the optimal solution to SINGLE-SUBSET-VERIFICATION is to check all but $l - 1$ of the votes for w (which will guarantee that the checked subset has at least two votes in common with every subset that requires verification, since these subsets contain $l + 1$ votes for w). In the next subsection, however, we study BOUNDED-SUBSET-VERIFICATION and SINGLE-SUBSET-VERIFICATION in their full generality.

4.2 ANALYZING THE COMPUTATIONAL PROBLEM

The BOUNDED-SUBSET-VERIFICATION and SINGLE-SUBSET-VERIFICATION problems are related to the (NP-complete) HITTING-SET [Karp, 1972] problem.

Definition 7 (HITTING-SET) *We are given a set T ($|T| = m$), a collection of subsets $\mathcal{U} = \{U_1, \ldots, U_n\}$ of T, and a number t. We are asked whether there exists a subset U' of T of size at most t such that for every $1 \le i \le n$, $|U_i \cap U'| \ge 1$.*

The definition of SINGLE-SUBSET-VERIFICATION is almost exactly the same as that of HITTING-SET: the only difference is that SINGLE-SUBSET-VERIFICATION requires each S_i to have at least *two* elements in common with S'. The special case of BOUNDED-SUBSET-VERIFICATION where $k = 2$—that is, we can only check subsets of size 2—is also closely related. We recall that for this special case, for each S_i, we need to check a subset (of size 2) of S_i. We can transform such an instance to a HITTING-SET instance as follows: let T be the set of all unordered pairs of elements of R, and let U_i be the set of all unordered pairs of elements of S_i. Then, a hitting set for T and $\mathcal{U}$ corresponds to a solution to the BOUNDED-SUBSET-VERIFICATION instance, and vice versa.

We are now ready to characterize the complexity of the problems that we are interested in. Even though we have formally defined the problems as decision problems, in a slight abuse of terminology, we will also discuss their approximability (for each problem, the objective is to minimize t). (The remaining proofs are omitted due to space constraint.)

Theorem 3 *Suppose that, under certain assumptions on complexity classes, HITTING-SET cannot be approximated to a ratio $\rho(m, n)$. Then, under the same assumptions, SINGLE-SUBSET-VERIFICATION cannot be approximated to ratio $\rho(m - 1, n)/2$.*

Theorem 4 *Suppose that, under certain assumptions on complexity classes, HITTING-SET cannot be approximated to a ratio $\rho(m, n)$. Then, under the same assumptions, BOUNDED-SUBSET-VERIFICATION with $k = 2$ cannot be approximated to ratio $\rho(m/2, n)$.*

It has been shown that HITTING-SET cannot be approximated to within a factor $(1 - o(1)) \ln(n)$ unless NP has quasi-polynomial time algorithms [Feige, 1998], so the results above imply similar logarithmic lower bounds on the approximation ratios that can be obtained on our problems. So, we have shown that negative results transfer from the HITTING-SET problem to our problems. On the other hand, the similarity of our problems to the HITTING-SET problem can also be used in a positive way, by adapting algorithms for the HITTING-SET problem to our problems. For example, the standard integer program for the HITTING-SET problem (**minimize** $\sum_{s \in T} x_s$ **subject to** for all $1 \le i \le n$, $\sum_{s \in U_i} x_s \ge 1$, $x_s \in \{0, 1\}$) is adapted to the SINGLE-SUBSET-VERIFICATION problem by simply changing the 1 in the constraint to a 2. Also, any optimal or approximation algorithm $A(T, \mathcal{U})$ for HITTING-SET can be turned into the following approximation algorithm $A'(R, \mathcal{S})$ for SINGLE-SUBSET-VERIFICATION.

1. Using A, compute a hitting set U' for the problem instance $(R, \mathcal{S})$.
2. Remove all the $S_i \in \mathcal{S}$ for which $S_i \cap U' \ge 2$ from the instance.
3. Remove all the elements in U' from the instance (both from R and from the elements of $\mathcal{S}$).
4. Compute a hitting set U'' for the remaining instance.
5. Return $U' \cup U''$.

Proposition 3 *If A always produces a $\rho(m, n)$-approximation to the HITTING-SET problem (if A is an optimal algorithm, then $\rho(m, n) = 1$), then A' always produces a $2\rho(m, n)$-approximation to the SINGLE-SUBSET-VERIFICATION problem.*

The bound in Proposition 3 can be shown to be just about tight. One famous greedy algorithm for HITTING-SET is to repeatedly choose the element that hits the most sets that were not hit previously; this algorithm obtains an $\ln n + 1$ approximation ratio [Johnson, 1974]. Thus, if we use this algorithm as A, then A' also obtains an $O(\log n)$ approximation algorithm for SINGLE-SUBSET-VERIFICATION, roughly matching the lower bound.

Any algorithm for HITTING-SET can also be used for BOUNDED-SUBSET-VERIFICATION with $k = 2$: as we described at the beginning of this subsection, an instance of BOUNDED-SUBSET-VERIFICATION with $k = 2$, given by $(R, \mathcal{S})$, can be transformed into the HITTING-SET instance where T is the set of all unordered pairs of elements in R, and U_i is the set of all unordered pairs of elements in S_i. Thus, a $\rho(m, n)$ approximation algorithm for HITTING-SET gives a $\rho(\binom{m}{2}, n)$ approximation for BOUNDED-SUBSET-VERIFICATION with $k = 2$. So, using the greedy algorithm again gives us an $O(\log n)$ approximation, roughly matching the lower bound.

5 CONCLUSIONS

In open, anonymous environments such as the Internet, mechanism design is complicated by the fact that a single agent can participate in the mechanism under multiple identifiers. One way to address this is to design *false-name-proof* mechanisms, which choose the outcome in such a way that agents have no incentive to use more than one identifier. Unfortunately, there are inherent limitations on what can be achieved with false-name-proof mechanisms, and at least in some cases, these limitations are crippling. An alternative approach is to verify the identities of all agents. This imposes significant overhead and removes any benefits from anonymity.

In this paper, we proposed a middle ground. Based on the reported preferences, we check, for various subsets of the reports, whether the reports in the subset were all submitted by different agents. If they were not, then we discard some of them. We characterized when such a limited verification protocol *induces* false-name-proofness for a mechanism, that is, when the combination of the mechanism and the verification protocol gives the agents no incentive to use multiple identifiers. The characterization is the following. A subset of the reports *requires verification* if an agent could have been better off submitting this subset instead of submitting a single, truthful report. Then, a protocol induces false-name-proofness if and only if every subset that requires verification has at least two reports in common with one of the checked subsets. This characterization leads to various optimization problems for minimizing verification effort given the minimal subsets that require verification. We studied the complexity of these problems, and showed that they are closely related to the NP-complete HITTING-SET problem, in the sense that negative as well as positive computational results for the HITTING-SET problem carry over to our problems. In the longer version of this paper, we also study how to efficiently determine the minimal subsets that require verification.

Many directions for future research remain. One such direction is to extend these concepts to mechanisms that are not strategy-proof (but, perhaps, Bayes-Nash incentive compatible). Another direction is to investigate *randomized* verification protocols: it should be possible to disincent false-name reporting by ensuring only that there is a *high probability* that some of a manipulating agent's reports will be checked.

References

Ed H. Clarke. Multipart pricing of public goods. *Public Choice*, 11:17–33, 1971.

Vincent Conitzer. Anonymity-proof voting rules, 2007. Draft.

Uriel Feige. A threshold of $\ln n$ for approximating set cover. *Journal of the ACM*, 45(4):634–652, 1998.

Allan Gibbard. Manipulation of voting schemes: a general result. *Econometrica*, 41:587–602, 1973.

Andrew Goldberg, Jason Hartline, and Andrew Wright. Competitive auctions and digital goods. *SODA*, 735–744, 2001.

Jerry Green and Jean-Jacques Laffont. Characterization of satisfactory mechanisms for the revelation of preferences for public goods. *Econometrica*, 45:427–438, 1977.

Theodore Groves. Incentives in teams. *Econometrica*, 41:617–631, 1973.

Johan Håstad. Clique is hard to approximate within $n^{1-\epsilon}$. *Acta Mathematica*, 182:105–142, 1999.

David Johnson. Approximation algorithms for combinatorial problems. *Journal of Computer and System Sciences*, 9:256–278, 1974.

Richard Karp. Reducibility among combinatorial problems. In Raymond E. Miller and James W. Thatcher, editors, *Complexity of Computer Computations*, 85–103. Plenum Press, NY, 1972.

Tokuro Matsuo, Takayuki Ito, Robert W. Day, and Toramatsu Shintani. A robust combinatorial auction mechanism against shill bidders. *AAMAS*, 1183–1190, 2006.

Roger Myerson. Incentive compatibility and the bargaining problem. *Econometrica*, 41(1), 1979.

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

Michael Rothkopf, Aleksandar Pekeč, and Ronald Harstad. Computationally manageable combinatorial auctions. *Management Science*, 44(8):1131–1147, 1998.

Tuomas Sandholm, David Levine, Michael Concordia, Paul Martyn, Rick Hughes, Jim Jacobs, and Dennis Begg. Changing the game in strategic sourcing at Procter & Gamble: Expressive competition enabled by optimization. *Interfaces*, 36(1):55–68, 2006.

Tuomas Sandholm. Algorithm for optimal winner determination in combinatorial auctions. *Artificial Intelligence*, 135:1–54, 2002.

Tuomas Sandholm. Expressive commerce and its application to sourcing. *IAAI*, 2006.

Mark Satterthwaite. Strategy-proofness and Arrow's conditions: Existence and correspondence theorems for voting procedures and social welfare functions. *Journal of Economic Theory*, 10:187–217, 1975.

William Vickrey. Counterspeculation, auctions, and competitive sealed tenders. *J. of Finance*, 16:8–37, 1961.

Makoto Yokoo, Yuko Sakurai, and Shigeo Matsubara. Robust combinatorial auction protocol against false-name bids. *Artificial Intelligence*, 130(2):167–181, 2001.

Makoto Yokoo, Yuko Sakurai, and Shigeo Matsubara. The effect of false-name bids in combinatorial auctions: New fraud in Internet auctions. *Games and Economic Behavior*, 46(1):174–188, 2004.

Makoto Yokoo, Toshihiro Matsutani, and Atsushi Iwasaki. False-name-proof combinatorial auction protocol: Groves mechanism with submodular approximation. *AAMAS*, 1135–1142, 2006.

Makoto Yokoo. The characterization of strategy/false-name proof combinatorial auction protocols: Price-oriented, rationing-free protocol. *IJCAI*, 733–742, 2003.