Novel Paradigm for the design of Obviously Strategyproof Mechanisms*
Extended Abstract

Diodato Ferraioli[1**], Adrian Meier[2], Paolo Penna[2], and Carmine Ventre[3]

[1] Università degli Studi di Salerno, Italy
dferraioli@unisa.it
[2] ETH Zurich, Switzerland
meiera@student.ethz.ch, paolo.penna@inf.ethz.ch
[3] King's College London, UK
carmine.ventre@kcl.ac.uk

Abstract. Taking care of the incentives of people with limited rationality is a challenging research direction that requires novel paradigms to design mechanisms. Obviously strategyproof (OSP) mechanisms have recently emerged as the concept of interest to this research agenda. However, the majority of the literature in the area has either highlighted the shortcomings of OSP or focused on the "right" definition rather than on the construction of these mechanisms.
We here give the first set of *tight* results on the approximation guarantee of OSP mechanisms for scheduling related machines and a *characterization* of set system instances for which optimal OSP mechanisms exist. By extending the well-known cycle monotonicity technique, we are able to concentrate on the algorithmic component of OSP mechanisms and provide some novel paradigms for their design. We prove that OSP encompasses careful interleaving of ascending and descending auctions.

1 Introduction

Mechanism design has been a very active research area that aims to develop algorithms (a.k.a., social choice functions) that align the objectives of the designer (e.g., optimality of the solution) with the incentives of self-interested agents (e.g., maximize their own utility).

One of the main obstacles to its application in realistic settings is the assumption of full rationality. Where theory predicts that people should not strategize, lab experiments show that they do (to their own disadvantage): this is, for example, the case for Vickrey's renown second-price auction; proved to be strategyproof and yet bidders lie when submitting sealed bids. However, lies are less

 Preliminary versions of the results herein have previously appeared in [5, 6].

** This author is partially supported by GNCS-INdAM and by the Italian MIUR PRIN 2017 Project ALGADIMAR "Algorithms, Games, and Digital Markets".

frequent when the same mechanism is implemented via an ascending auction [10].

A vague explanation of this phenomenon is that, from the point of view of a bidder, the strategyproofness (a.k.a., truthfulness) of an ascending price auction is *easier to grasp* than the strategyproofness of the second-price sealed bid auction [2]. The key difference here is the way these two auctions are *implemented*:

– In the second-price sealed-bid auction (direct-revelation implementation), each bidder submits her own bid *once* (either her true valuation or a different value). This mechanism is *strategyproof* meaning that truth-telling is a dominant strategy: *for every report of the other bidders*, the utility when truth-telling is not worse than the utility when bidding untruthfully.
– In the ascending price auction (extensive-form implementation), each bidder is repeatedly offered some price that she can accept (stay in the auction) or reject (leave the auction). In this auction, momentarily *accepting a good price* guarantees a non-negative utility, while *rejecting a good price* or *accepting a bad price* yield non-positive utility. Here good price refers to the private valuation of the bidder and, intuitively, truth-telling in this auction means accepting prices as long as they are not above the true valuation.

Intuitively speaking, in the second type of auction, it is *obvious* for a bidder to decide her strategy, because the utility for the *worst scenario* when truth-telling is at least as good as that of the *best scenario* when cheating. The recent definition of *obviously strategyproof (OSP)* mechanisms [12] formalizes this argument: ascending auctions are OSP mechanisms, while sealed-bid auctions are not. Interestingly, [12] proves that a mechanism is OSP *if and only if* truth-telling is dominant even for bidders who lack contingent reasoning skills, thus addressing a specific form of bounded rationality.

As being OSP is *stronger* than being strategyproof, it is natural to ask if this has an impact on what can be done by such mechanisms. For instance, the so-called *deferred-acceptance* (DA) auctions [14] are OSP (as they essentially are ascending price auctions), but unfortunately their performance (approximation guarantee) for several optimization problems is quite poor compared to what strategyproof mechanisms can do [4]. Whether this is an inherent limitation of OSP mechanisms or just of this technique is not clear.

One of the reasons behind this open question might be the absence of a general technique for designing OSP mechanisms and the lack of an algorithmic understanding of OSP mechanisms. Specifically, it is well known that strategyproofness is equivalent to certain *monotonicity* conditions of the *algorithm* used by the mechanism for computing the solution (be it an allocation of goods or a path in a network with self-interested agents). Therefore, one can essentially focus on the algorithmic part and study questions regarding the quality of the solutions (e.g., approximation) and the time needed to compute a solution (e.g., complexity). The same type of questions seem much more challenging for OSP mechanisms, as such characterizations are not known. Recent work in the area, such as [1, 3, 13], mainly attempts to simplify the notion of OSP, whilst [15, 16, 7, 8] define, among other results, stronger and weaker versions of OSP.

The goal of this work is to build the foundations to reason about OSP algorithmically. In particular, we advance the state of the art by providing an algorithmic characterization of OSP mechanisms. Among others, our results show why deferred acceptance auctions [14] – essentially the only known technique to design OSP mechanisms with money – do not fully capture the power of a "generic" OSP mechanism, as the latter may exploit some aspects of the implementation (i.e., extensive-form game) in a crucial way.

Our Contribution. To give an algorithmic characterization of OSP mechanisms, we extend the well known cycle-monotonicity (CMON) technique. This approach allows to abstract the truthfulness of an algorithm in terms of non-negative weight cycles on suitably defined graphs. In its more general form, the graph for truthfulness of agent i is complete and has a vertex for each possible outcome; edge weights account for the incentives of agent i in forcing a different outcome by lying. We show that non-negative weight cycles continue to characterize OSP when the graph of interest is carefully defined. Specifically, we have a graph for each agent i as well, but there is a node for each strategy profile and we add an edge between two vertices only when there is an OSP constraint for i involving those profiles. Essentially, our main conceptual contribution is a way to accommodate the OSP constraints, which *depend on the particular extensive-form implementation* of the mechanism, in the machinery of CMON, which is designed to focus on the algorithmic output of mechanism. We prove that payments exist that can be paired to the algorithm in an OSP mechanism, implemented in extensive-form, if and only if this graph does not contain negative cycles.

Interestingly, our technique shows the interplay between algorithms (which outcome/solution to return) and how the mechanism is implemented as an extensive-form game (what we call the *implementation tree*). Roughly speaking, our characterization says which algorithms can be used for *any* choice of the implementation tree. The ability to choose between different implementation trees is what gives extra power to the designer: for example, the construction of OSP mechanisms based on DA auctions [14] always uses *the same* fixed tree for all problems and instances. Though this yields a simple algorithmic condition, it can be wasteful in terms of optimality (approximation guarantee) as we show herein. In fact, for our results, we will use CMON *two ways* to characterize both algorithmic properties (having fixed an implementation tree) and implementation properties (having fixed the approximation guarantee we want to achieve).

Armed with the OSP CMON technique, we are able to give the first *tight* bounds on the approximation ratio of OSP mechanisms for the problem of scheduling n related machines (for identical jobs) and a *characterization* of optimal OSP mechanisms for set system problems (which include path auctions as a special case). A caveat for our results is about the size of the agents' domains. While our lower bounds/necessary conditions hold regardless of the size of the domain, the mechanisms that we provide are shown to be OSP only for two- and three-value (agent-specific) domains, as we prove that these are the only cases in which non-negative two-cycles are necessary and sufficient. However, our mechanisms are, to the best of our knowledge, the first examples of OSP mechanisms with money that do not follow a clock or a posted price auction format (other mechanisms that do not follow these formats have been proposed only for setting without money, namely matching and voting [12, 1, 3, 15]). One of the main messages of our work is exactly that it is possible to combine ascending and descending phases for the implementation trees of algorithms with good approximation guarantees and obtain OSP mechanisms.

Machine Scheduling. Machine scheduling is one of the problems that received most attentions within the literature about OSP. In particular, [7] provides a *constant* lower bound for the approximation ratio achievable by OSP mechanisms for machine scheduling when payments have a specific structure (as discussed below, here we improve this bound to $\sqrt{n}$ by using the CMON characterization of OSP). They also provide an upper bound by using *monitoring*, a model wherein agents pay their reported costs (instead of their actual cost, as it is instead assumed in our work). Monitoring is also used in [11] to prove a tight bound for OSP mechanisms without money and a single task. The tradeoff between approximation guarantee and relaxations of OSP is recently studied in [9].

In this work, we show that the optimum for machine scheduling can be implemented OSP-ly when the agents' domains have size two. We prove that given a "balanced" optimum (i.e., a greedy allocation of jobs to machines) we can always find an implementation tree for which OSP is guaranteed. The mechanism directly asks the queried agents to reveal their type; given that the domain only contains two values, this is basically a descending/ascending auction.

For domains of size three, instead, we give a lower bound of $\sqrt{n}$ and an essentially tight upper bound of $\lceil\sqrt{n}\rceil$. Interestingly, the latter is proved with two different OSP mechanisms – one assuming more than $\lceil\sqrt{n}\rceil^2$ number of jobs and the second under the hypothesis that there are less than that.

> **Main Theorem about Machine Scheduling (informal).** *The tight approximation guarantee of OSP mechanisms that can be guaranteed over all three-value domains is $\sqrt{n}$. The OSP mechanisms use a descending auction (to find the $n - \lceil\sqrt{n}\rceil$ slowest machines) followed by an ascending auction (to find the fastest machine(s)).*

On the technical level, these results are shown by using our approach of CMON two ways. We prove that any better than $\sqrt{n}$-approximate OSP mechanism must have the following structure: for a number of rounds, the mechanism must (i) separate, in its implementation tree, the largest and the second largest value in the domain; (ii) assign nothing to agents who have maximum value in the domain. The former property restricts the family of implementation trees we can use, whilst the latter restricts the algorithmic output. Our lower bound shows that there is nothing in this intersection.

Our matching upper bounds need to find *both* the implementation tree and the algorithm satisfying OSP and the approximation guarantee. While the general idea of the implementation is that of a descending auction followed by an ascending auction independently of the number of jobs, we need to tailor the design of the mechanisms (namely, their ascending phase) according to the number of jobs to achieve OSP and the desired approximation simultaneously. This proves two important points. On one hand, the design of OSP mechanisms is challenging yet interesting as one needs to carefully balance algorithms and their implementation. On the other hand, it proves why fixing the implementation, as in DA auctions, might be the wrong choice. We in fact extend and adapt our analysis to prove that any ascending and descending (thus including DA) auction has an approximation of n.

Set Systems. We consider a general set system problem, wherein agents have three-value (heterogeneous) domains and fully characterize the properties needed to design OSP mechanisms.

> **Main Theorem about Set Systems (informal).** *There is an OSP optimal mechanism iff the set of feasible solutions are "aligned" with agents' subdomains. The OSP optimal mechanism, if any, combines ascending and descending auctions depending on the structure of the feasible solution set.*

The intuition behind the characterization is simple. From OSP CMON, we know that if an OSP mechanism selects an agent e when she has a "high" cost, then it must select e when she has a "low" cost (akin to monotonicity for strategyproofness). Therefore, to design an OSP optimal mechanism, we need to define an implementation tree which satisfies this property. At each node of the tree, the domain of the agents is restricted to a particular subdomain, depending on the particular history; in turn, the set of possible type profiles also shrinks. Hence, there may be solutions that become suboptimal for all type profiles in this set, and others that are still *alive* (i.e., optimal for at least one type profile in the set). When e is asked to separate a high cost from a low cost at node u of the tree, we then need the alive solutions to be "aligned" for the subdomain at u, which roughly means that it should never be the case that there are two bid profiles in this subdomain for which e belongs to an optimal solution when she has a high cost and is not part of an optimal solution when she has a low cost.

The somehow surprising extra aspect is that, even if the alive solutions were not aligned for one single subdomain, then there would be no way to design an implementation tree to bypass this misalignment.

The technical definition of alignment has some nuisance to do with the particular ways in which the OSP monotonicity can be broken, but on the positive side, rather immediately suggests how to interleave ascending and descending phases to design an OSP optimal mechanism. This characterization precisely shows how OSP needs to look at the quality of solutions among set of instances (encoded by agent subdomains) rather than just the single instance and how this is needed to inform the shape of the implementation tree.

Future Directions. A technical one is about the domain size and the difference between 2-cycles and longer ones; to what extent adding an extra type in the domain can deteriorate the approximation ratio of OSP mechanisms? A second, more conceptual question, is about dealing with multi-parameter agents. Indeed, it does not seem immediate to characterize the implementation trees for this kind of agents as there is not a concept of relative ordering of types.

References

1. I. Ashlagi and Y. A. Gonczarowski. Stable matching mechanisms are not obviously strategy-proof. *J. Economic Theory*, 177:405–425, 2018.
2. L. M. Ausubel. An efficient ascending-bid auction for multiple objects. *American Economic Review*, 94(5):1452–1475, 2004.
3. S. Bade and Y. A. Gonczarowski. Gibbard-Satterthwaite success stories and obvious strategyproofness. In *EC 2017*, page 565, 2017.
4. P. Dütting, V. Gkatzelis, and T. Roughgarden. The performance of deferred-acceptance auctions. *Math. Oper. Res.*, 42(4), 2017.
5. D. Ferraioli, A. Meier, P. Penna, and C. Ventre. Automated optimal osp mechanisms for set systems: The case of small domains. In *WINE 2019*, 2019.
6. D. Ferraioli, A. Meier, P. Penna, and C. Ventre. Obviously strategyproof mechanisms for machine scheduling. In *ESA 2019*, 2019.
7. D. Ferraioli and C. Ventre. Obvious strategyproofness needs monitoring for good approximations. In *AAAI 2017*, pages 516–522, 2017.
8. D. Ferraioli and C. Ventre. Probabilistic verification for obviously strategyproof mechanisms. In *IJCAI 2018*, 2018.
9. D. Ferraioli and C. Ventre. Obvious strategyproofness, bounded rationality and approximation: The case of machine scheduling. In *SAGT 2019*, 2019.
10. J. Kagel, R. Harstad, and D. Levin. Information impact and allocation rules in auctions with affiliated private values: A laboratory study. *Econometrica*, pages 1275–1304, 1987.
11. M. Kyropoulou and C. Ventre. Obviously strategyproof mechanisms without money for scheduling. In *AAMAS 2019*, 2019.
12. S. Li. Obviously strategy-proof mechanisms. *American Economic Review*, 107(11):3257–87, 2017.
13. A. Mackenzie. A revelation principle for obviously strategy-proof implementation. Research Memorandum 014, (GSBE), 2017.
14. P. Milgrom and I. Segal. Deferred-acceptance auctions and radio spectrum reallocation. In *EC 2014*, 2014.
15. M. Pycia and P. Troyan. Obvious dominance and random priority. In *EC 2019*, 2019.
16. L. Zhang and D. Levin. Bounded rationality and robust mechanism design: An axiomatic approach. *American Economic Review*, 107(5):235–39, 2017.