Obvious Strategyproofness, Bounded Rationality and Approximation

Extended Abstract

Diodato Ferraioli
Università degli Studi di Salerno, Italy
dferraioli@unisa.it

Carmine Ventre
University of Essex, UK
c.ventre@essex.ac.uk

ABSTRACT

Obvious strategyproofness (OSP) has recently emerged as the solution concept of interest to study incentive compatibility in presence of agents with a specific form of bounded rationality, i.e., those who have *no* contingent reasoning skill whatsoever. We here want to study the relationship between the approximation guarantee of incentive-compatible mechanisms and the *degree* of rationality of the agents, intuitively measured in terms of the number of contingencies that they can handle in their reasoning. We weaken the definition of OSP to accommodate for cleverer agents and study the trade-off between approximation and agents' rationality for the paradigmatic machine scheduling problem. We prove that, at least for the classical machine scheduling problem, "good" approximations are possible if and only if the agents' rationality allows for a significant number of contingencies to be considered, thus showing that OSP is not too restrictive a notion of bounded rationality from the point of view of approximation.

ACM Reference Format:
Diodato Ferraioli and Carmine Ventre. 2019. Obvious Strategyproofness, Bounded Rationality and Approximation. In *Proc. of the 18th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2019), Montreal, Canada, May 13–17, 2019,* IFAAMAS, 3 pages.

1 INTRODUCTION

Mechanism design is an established research field, by now rooted in a number of academic disciplines including theoretical computer science and AI. Its main objective is that of computing in presence of selfish agents who might misguide the designer's algorithm if it is profitable for them to do so. The concept of *strategyproofness* (SP-ness) (a.k.a., *truthfulness*) ensures that the algorithm and the agents' incentives are compatible and computation is indeed viable.

SP is based on the assumption of full rationality: agents are able to consider all possible strategies and their combinations to reason about their incentives. Nevertheless, this assumption is seldom true in reality and it is often the case that people strategize against mechanisms that are known to be truthful [3]. One then needs a different notion to compute in the presence of agents with bounded rationality. The problem here is twofold: how can we formalize strategyproofness for agents with (some kind of) bounded rationality? If so, can we *quantify* this bounded rationality and relate that to the *performances* of the mechanisms?

Proc. of the 18th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2019), N. Agmon, M. E. Taylor, E. Elkind, M. Veloso (eds.), May 13–17, 2019, Montreal, Canada.

The first question has been recently addressed by Li [9], who defines the concept of *obvious strategyproofness* (OSP-ness); this notion has attracted quite a lot of interest in the community [2, 4, 6–8, 10, 12, 14]. Here, the mechanism is seen as an extensive-form game; when a decision upon the strategy to play has to be made, it is assumed that the reasoning of each agent i is as simple as the following: the *worst* possible outcome that she can get when behaving well (this typically corresponds to playing the game according to the so-called agent's true *type*) must be at least as good as the *best* outcome when misbehaving (that is, following a different strategy). Best/Worst are quantified over *all* the possible strategies that the players playing in the game after i can adopt. Li [9] proves that this is the right solution concept for a model of bounded rationality wherein agents have *no* contingent reasoning skills; rather than thinking about the possible if-then-else's, an agent is guaranteed that honesty is the best strategy no matter all the contingencies.

Given the OSP formalization of bounded rationality, we focus, in this work, on the second question. On the one hand, OSP is too restrictive in that people might be able, within their computational limitations, to consider few cases of if-then-else's. On the other hand, OSP mechanisms appear to be quite limited, with respect to SP ones, in terms of their approximation guarantee [6, 7]. The question then becomes:

> *Can we quantify the trade-off between the "degree" of bounded rationality of the agents and the approximation guarantee of the mechanisms incentivizing them?*

Our Contribution. The concept of *lookahead* is discussed in the literature in the context of (strategies to play) games, and agents with limited computational capabilities. De Groot [5] found that all chess players (of whatever standard) used essentially the same thought process – one based upon a lookahead heuristic. Shannon [13] formally proposed the lookahead method and considered it a practical way for machines to tackle complex problems, whilst, in his classical book on heuristic search, Pearl [11] described lookahead as the technique being used by "almost all game-playing programs".

We propose to consider lookahead as a way to quantify bounded rationality, in relation to OSP. Whilst in OSP the players have no lookahead at all, we here consider the case in which the agents have lookahead k, k going from 0 (OSP) to $n-1$ (SP). Intuitively, k measures the number of players upon which each player reasons about in her decision making. So when agent i has to decide upon the strategy to play, she will consider all the possible cases (strategies) for these k agents (à la SP) and a no-contingent reasoning (à la OSP) for the others. We regard our definition of *OSP with k-lookahead* (k-OSP, for short) as a major conceptual contribution of our work.

We then look at the trade-off between the value of k and the approximation guarantee of k-OSP mechanisms. We focus of the well-studied problem of machine scheduling, where n agents control related machines and the objective is to schedule a set of m (identical) jobs to the machines so to minimize the *makespan* (i.e., the latest machine's completion time). In our main technical contribution, we prove a lower bound on approximation guarantee of $\tau_k(n) = \frac{\sqrt{k^2+4n}-k}{2}$, thus providing a smooth transition function between the known approximation factors of $\sqrt{n}$ for OSP mechanisms [6] and 1 for SP mechanisms [1].

The main message of our work is that having more rational agents only slightly improves the approximation guarantee of incentive-compatible mechanisms, at least in the case of machine scheduling. In fact, to have a constant approximation of the optimum makespan one would need agents with a $\omega(1)$-lookahead. We can then conclude that, in the cases in which the agents are not that rational, OSP is not that restrictive a solution concept to study the approximation of mechanisms for agents with bounded rationality.

2 THE DEFINITION

We have a set N of n agents; each agent i has a domain D_i of possible *types* – encoding some feature of theirs (e.g., their speed). The actual type of agent i in D_i is her private knowledge. An extensive-form mechanism $\mathcal{M}$ is a triple $(f, p, \mathcal{T})$, where f is an algorithm that takes as input bid profiles and returns a feasible solution, $p = (p_1, \ldots, p_n)$ is the payment function, one for each agent, and $\mathcal{T}$ is an extensive-form game, that we call *implementation tree*. Intuitively, $\mathcal{T}$ represents the steps that the mechanism will take to determine its outcome. More formally, each internal node u of $\mathcal{T}$ is labelled with a player $S(u)$, called the *divergent agent* at u, and the outgoing edges from u are labelled with types in the domain of $S(u)$ that are compatible with the history leading to u; the edge labels denote a partition of the compatible types. We denote by $D_i(u)$ the types in the domain of i that are compatible with the history leading to node $u \in \mathcal{T}$. The tree models how $\mathcal{M}$ interacts with the agents: at node u the agent $S(u)$ is queried and asked to choose an action, that corresponds to selecting one of u's outgoing edges. The chosen action *signals* that the type of $S(u)$ is in the set of types labeling the corresponding edge. The leaves of the tree will then be linked to (a set of) bid profiles; the mechanism will return (f, p) accordingly; in other words, each leaf corresponds to an outcome of the mechanism. (Observe that this means that the domain of f and p is effectively given by the leaves of $\mathcal{T}$.) We use $\mathbf{b}$ to denote bid profiles, so that b_i stands for the type that i signalled to the mechanism. For simplicity, we use $f(\mathbf{b})$ and $p_1(\mathbf{b}), \ldots, p_n(\mathbf{b})$ to denote the outcome of (f, p) for the leaf of $\mathcal{T}$ to which $\mathbf{b}$ belongs. We assume that agents have quasi-linear utilities, that is, agent i of type t who signals (i.e., plays the game $\mathcal{T}$ according to) b has utility $u_i(b, \mathbf{b}_{-i}) = p_i(\mathbf{b}) - t(f(\mathbf{b}))$, where, with a slight abuse of notation, $t(f(\mathbf{b}))$ is the cost that player i pays to implement the outcome $f(\mathbf{b})$ when her type is t, and $\mathbf{b}_{-i}$ is the declaration vector of all agents except i. (More generally, given a set $A \subset N$, we let $\mathbf{b}_A = (b_j)_{j \in A}$.)

Figure 1 gives an example of an implementation tree where three players have domain $\{L, H\}$. The root partitions the domain of machine 1 into L and H. If v is the left child, then $D_1(v) = \{L\}$ as type H is no longer compatible with the history of v.

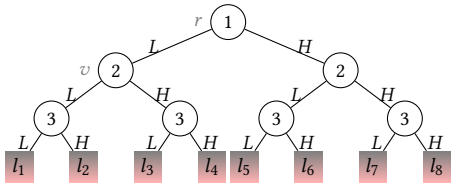

Figure 1: An implementation tree with three players with two-value domains $\{L, H\}$; each player separates the domain types upon playing; at each leaf l_i the mechanism computes $f(\mathbf{b})$ and $p(\mathbf{b})$, $\mathbf{b}$ being the bid vector at l_i.

We call a bid profile $\mathbf{b}$ *compatible with* u if $\mathbf{b}$ is compatible with the history of u for all agents. We furthermore say that $(t, \mathbf{b}_{-i})$ and $(b, \mathbf{b}'_{-i})$ diverge at u if $i = S(u)$ and t and b are labels of different edges outgoing u (we sometimes will abuse notation and we also say that t and b diverge at u). So, for example, (L, H, H) and (L, L, H) are compatible at node v on Figure 1 and diverge at that node, whilst (L, L, H) and (L, L, L) are compatible but do not diverge.

For every agent i and types $t, b \in D_i$, we let $u^i_{t,b}$ denote a vertex u in the implementation tree $\mathcal{T}$, such that $(t, \mathbf{b}_{-i})$ and $(b, \mathbf{b}'_{-i})$ are compatible with u, but diverge at u for some $\mathbf{b}_{-i}, \mathbf{b}'_{-i} \in D_{-i}(u) = \times_{j \neq i} D_j(u)$. We finally denote i's lookahead at $u^i_{t,b}$ as $L_k(u^i_{t,b})$, that is, a set of (at most) k agents that move in $\mathcal{T}$ after i.

Definition 2.1 (OSP with k-lookahead). An extensive-form mechanism $\mathcal{M} = (f, \mathcal{T}, p)$ is OSP with k-lookahead (k-OSP, for short) if for all i, $t, b \in D_i$, t being i's type, $u^i_{t,b} \in \mathcal{T}$, $\mathbf{b}_K \in D_K(u^i_{t,b})$ and $\mathbf{b}_T, \mathbf{b}'_T \in D_T(u^i_{t,b})$, it holds that $u_i(t, \mathbf{b}_K, \mathbf{b}_T) \geq u_i(b, \mathbf{b}_K, \mathbf{b}'_T)$, where $K = L_k(u^i_{t,b})$, $T = N \setminus (K \cup \{i\})$ and $D_A(u) = \times_{j \in A \subset N} D_j(u)$.

Essentially, a mechanism is OSP with lookahed if each agent is willing to behave truthfully at each node of the tree in which she interacts with the mechanism, provided that she exactly knows the types of agents in K, but has no information about agents in T, except that their types are compatible with the history.

We remark that with $k = 0$ we get the definition of OSP – wherein K is empty – and with $k = n - 1$ we have truthfulness.

3 THE CASE OF MACHINE SCHEDULING

We are given a set of m identical jobs and the n agents control related machines. Agent i's type is a job-independent processing time t_i per unit of job. The algorithm f must choose a possible schedule $f(\mathbf{b}) = (f_1(\mathbf{b}), \ldots, f_n(\mathbf{b}))$ of jobs to the machines, where $f_i(\mathbf{b})$ denotes the job load assigned to machine i when agents take actions signalling $\mathbf{b}$. The cost that agent i faces for the schedule $f(\mathbf{b})$ is $t_i(f(\mathbf{b})) = t_i \cdot f_i(\mathbf{b})$. We focus on algorithms f^* minimizing the *makespan*, i.e., $f^*(\mathbf{b}) \in \arg\min_{\mathbf{x}} \max_{i=1}^n b_i(\mathbf{x})$. We say that f is α-approximate if it returns a solution whose cost is a factor α away from the optimum.

Let $\tau_k(n) = \frac{\sqrt{k^2+4n}-k}{2}$. That is, τ_k is a function of n such that $n = \tau_k(n)(\tau_k(n) + k)$. Observe that $\tau_0(n) = \sqrt{n}$ and $\tau_{n-1}(n) = 1$. We can then prove the following theorem.

Theorem 3.1. *For the machine scheduling problem, no k-OSP mechanism can be better than $\tau_k(n)$-approximate. This even holds for homogeneous three-value domains, i.e., $D_i = \{L, M, H\}$ for every i.*

REFERENCES

[1] Aaron Archer and Éva Tardos. 2001. Truthful Mechanisms for One-Parameter Agents. In *FOCS 2001*. 482–491.

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

[3] Lawrence M. Ausubel. 2004. An efficient ascending-bid auction for multiple objects. *American Economic Review* 94, 5 (2004), 1452–1475.

[4] Sophie Bade and Yannai A. Gonczarowski. 2017. Gibbard-Satterthwaite Success Stories and Obvious Strategyproofness. In *EC 2017*. 565.

[5] Adriaan De Groot. 1978. *Thought and Choice in Chess*. Mouton.

[6] Diodato Ferraioli, Adrian Meier, Paolo Penna, and Carmine Ventre. 2018. On the approximation guarantee of obviously strategyproof mechanisms. *CoRR* abs/1805.04190 (2018).

[7] Diodato Ferraioli and Carmine Ventre. 2017. Obvious Strategyproofness Needs Monitoring for Good Approximations. In *AAAI 2017*. 516–522.

[8] Diodato Ferraioli and Carmine Ventre. 2018. Probabilistic Verification for Obviously Strategyproof Mechanisms. In *IJCAI 2018*.

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

[10] Andrew Mackenzie. 2017. A revelation principle for obviously strategy-proof implementation. (2017).

[11] Judea Pearl. 1984. *Heuristics: Intelligent Search Strategies for Computer Problem Solving*. Addison-Wesley.

[12] Marek Pycia and Peter Troyan. 2016. Obvious dominance and random priority. (2016).

[13] Claude Shannon. 1950. Programming a computer for playing chess. *Philos. Mag.* 41, 314 (1950), 256–275.

[14] Luyao Zhang and Dan Levin. 2017. Bounded rationality and robust mechanism design: An axiomatic approach. *American Economic Review* 107, 5 (2017), 235–39.