The Pareto Frontier of Inefficiency in Mechanism Design*

Aris Filos-Ratsikas† Yiannis Giannakopoulos‡ Philip Lazos§

September 10, 2018

Abstract

We study the trade-off between the Price of Anarchy (PoA) and the Price of Stability (PoS) in mechanism design, in the prototypical problem of unrelated machine scheduling. We give bounds on the space of feasible mechanisms with respect to the above metrics, and observe that two fundamental mechanisms, namely the First-Price (FP) and the Second-Price (SP), lie on the two opposite extrema of this boundary. Furthermore, for the natural class of anonymous task-independent mechanisms, we completely characterize the PoA/PoS Pareto frontier; we design a class of optimal mechanisms $\mathcal{SP}_\alpha$ that lie *exactly* on this frontier. In particular, these mechanisms range smoothly, with respect to parameter $\alpha \geq 1$ across the frontier, between the First-Price ($\mathcal{SP}_1$) and Second-Price ($\mathcal{SP}_\infty$) mechanisms.

1 Introduction

The field of *algorithmic mechanism design* was established in the seminal paper of Nisan and Ronen [35] and has ever since been at the centre of research in the intersection of economics and computer science. The research agenda put forward in [35] advocates the study of approximate solutions to interesting optimization problems, in settings where rational agents are in control of the input parameters. More concretely, the authors of [35] proposed a framework in which, not unlike classical approaches in approximation algorithms, algorithms that operate under certain limitations are evaluated in terms of their approximation ratio. In particular, in algorithmic mechanism design, this constraint comes from the requirement that agents should have the right incentives to always report their inputs *truthfully*. The corresponding algorithms, paired with appropriately chosen payment functions, are called *mechanisms* [34].

Another pioneering line of work, initiated by Koutsoupias and Papadimitriou [24] and popularized further by Roughgarden and Tardos [38], studies the *inefficiency* of games through the notion of the *Price of Anarchy* (PoA), which measures the deterioration of some objective at the worst-case Nash equilibrium. A more optimistic version of the same principle, where the inefficiency is measured at the *best* equilibrium [41], was introduced by Anshelevich et al. [1] under the name of *Price of Stability* (PoS).

Given the straightforward observation that mechanisms induce games between the agents that control their inputs, as well as the fact that truthfulness is typically a very demanding property, an alternative approach to the framework of Nisan and Ronen [35] is to design mechanisms that perform well *in the equilibrium*, i.e., they provide good PoA or PoS guarantees.

*Supported by ERC Advanced Grant 321171 (ALGAME), the Swiss National Science Foundation under contract No. 200021_165522 and the Alexander von Humboldt Foundation with funds from the German Federal Ministry of Education and Research (BMBF).

†École Polytechnique Fédérale de Lausanne. Email: aris.filosratsikas@epfl.ch

‡TU Munich. Email: yiannis.giannakopoulos@tum.de

§University of Oxford. Email: philip.lazos@cs.ox.ac.uk

This approach has been adopted, among others, by central papers in the field (e.g., see [42, 39] and references therein) and is by now as much a part of algorithmic mechanism design as the original framework of [35].

While the literature that studies the concepts of PoA and PoS is long and extensive, there seems to be a lack of a *systematic approach* investigating the trade-off between the two notions *simultaneously*. More concretely, given a problem in algorithmic mechanism design, it seems quite natural to explore not only the best mechanisms in terms of the two notions independently, but also the mechanisms that achieve the best trade-off between the two. In a sense, this approach concerns a "tighter" optimality notion, as among a set of mechanisms with an "acceptable" Price of Anarchy guarantee, we would like to identify the ones that provide the best possible Price of Stability. Our contribution in the current paper is the proposal of such a research agenda and its application on the canonical problem in the field, introduced in the seminal work of Nisan and Ronen [35], that of scheduling on unrelated machines.

1.1 Our contributions

We propose the *research agenda of studying systematically the trade-off between the Price of Anarchy and the Price of Stability in algorithmic mechanism design.* Specifically, given a problem at hand and an objective function, we are interested in the trade-off between the PoA and the PoS of mechanisms for the given objective. We apply this approach on the prototypical problem of algorithmic mechanism design studied in [35], that of unrelated machine scheduling, where the machines are self-interested agents.

First, in Section 3, for the class of *all* possible mechanisms, we prove that PoA guarantees imply corresponding PoS lower bounds and vice-versa (Theorem 2), which allows us to quantify the possible trade-off between the two inefficiency notions in terms of a feasible region (see Fig. 2); we refer to the the boundary of this region as the *inefficiency boundary*. Interestingly, adaptations of two well-known mechanisms, the First-Price and the Second-Price mechanisms, turn out to lie on the extreme points of this boundary.

Next, in Section 4, for the well-studied class of task-independent and anonymous mechanisms,[1] we are able to show a tighter feasibility region (Theorem 6). As a matter of fact, its inefficiency boundary turns out to *completely characterize* the achievable trade-off between the PoA and the PoS: we design a class of mechanisms (Section 4.2) called $\mathcal{SP}_\alpha$, parameterized by a quantity α, which are *optimal* in the sense that for any possible trade-off between the two inefficiency notions, there exists a mechanism in the class (i.e., an appropriate choice of α) that exactly achieves this trade-off (Theorems 7 and 8). In other words, we obtain an exact description of the *Pareto frontier of inefficiency* (see Fig. 3).

Our $\mathcal{SP}_\alpha$ mechanisms are simple and intuitive and are based on the idea of setting reserve prices *relatively* to the declarations of the fastest machines. While this is clearly not truthful, we prove that it induces the equilibria which are desirable for our results. More precisely, the choice of α enables us to "control" the set of possible equilibria in a way that allows us to achieve any trade-off on the boundary.

Our results also offer insights in an other interesting direction. The inefficiency boundary result for general mechanisms is based on a novel monotonicity lemma (Lemma 1), which is quite different from the well-known *weak monotonicity* property [40], (e.g., see [35, 9]). Interestingly, we also use this lemma to prove a general lower bound of n on the PoA of *any* mechanism for the scheduling problem (Theorem 1), where n is the number of machines. This result contributes to the interesting debate [23, 20, 11] of whether general mechanisms (that may be non-truthful)

[1]We remark that the best known mechanisms for several variants of truthful scheduling are task-independent and anonymous; see also Section 5 for a more detailed discussion.

can outperform truthful ones, providing a negative answer (see also Section 1.2.2). As a matter of fact, in Theorem 5, we actually show that when evaluated at their worst-case equilibrium, truthful mechanisms are bound to perform poorly, as their PoA is unbounded.

Finally, in Section 5, we conclude with a thorough discussion, where we identify several intriguing directions for future work, both on a technical and a conceptual level.

1.2 Related Work

1.2.1 The (Selfish) Scheduling Problem

The scheduling problem on unrelated machines with selfish participants is the prototypical problem studied by Nisan and Ronen [35] in 1999, when they introduced the field of algorithmic mechanism design. The authors consider the worst-case performance of truthful mechanisms on dominant strategy, truth-telling equilibria, and identify that an adaptation of the well-known Second-Price auction[2] has an approximation ratio of n for the problem, where n is the number of machines. Despite several attempts over the years, this is still the best-known truthful mechanism. On the other hand, the succession of the best proven lower bounds started with 2 in [35], improved to 2.41 in [25] and finally to 2.61 in [9]. Interestingly, Ashlagi et al. [3] showed a matching lower bound of n for *anonymous* mechanisms (i.e., mechanisms that do not take the identities of the machines into account) and whether there is a better mechanism that is not anonymous is still the most prominent open problem in the are. In any case, anonymity is in general a desirable property which is satisfied by most natural mechanisms, including the best possible mechanisms for scheduling; we further discuss the role of this property in our setting in Section 5.2.

Several other variants of the problem have also been considered over the years, such as randomized mechanisms [35, 28, 31], fractional scheduling [10], Bayesian scheduling [6, 13, 19] or restricted domains where the processing times come from discrete sets [26]. Alongside the approximation ratio results, there has also been work on structural properties and characterizations [15, 8]. For a more detailed exposition of some of these results, we refer the reader to the survey of Christodoulou and Koutsoupias [7].

1.2.2 The Truthful Setting vs. the Strategic Setting

As we mentioned earlier, given that truthfulness is a very demanding requirement which imposes strict constraints on the allocation and payment functions, it is an interesting direction to consider whether *non-truthful* mechanisms could perform better, when evaluated in the worst-case equilibrium. In other words, for a given problem, one could ask the following question:

> *"Do there exist (non-truthful) mechanisms whose Price of Anarchy outperforms the approximation ratio guarantee of all truthful mechanisms?"*.

To differentiate, we will refer to the traditional approach of Nisan and Ronen [35] as the *truthful setting* and to the setting where all mechanisms are explored (with respect to their Nash equilibria) as the *strategic setting*.

Koutsoupias [23] studied the truthful setting for the problem of unrelated machine scheduling *without money* but he explicitly advocated the strategic setting as a future direction. This was later pursued by Giannakopoulos et al. [20] for the same problem, where the authors answered the aforementioned question in the affirmative. The same approach was taken by Christodoulou

[2] In the related literature, this mechanism is often referred to as the Vickrey-Clarke-Groves (VCG) mechanism [43, 12, 21]. The mechanism was originally referred to as the "minWork Mechanism" in [35].

et al. [11] following the results of Filos-Ratsikas et al. [17] on the limitations of truthful mechanisms for indivisible item allocation. In the literature of auctions, the strategic setting was studied even in domains for which an optimal truthful mechanism (the VCG mechanism) exists, motivated by the fact that non-truthful mechanisms are being employed in practice, with the Generalized Second-Price auction used by Google for the Adwords allocation being a prominent example [5]. We refer the reader to the survey of Roughgarden et al. [39] for more details.

Somewhat surprisingly, although the exploration of different solution concepts besides dominant-strategy equilibria was already explicitly mentioned as a future direction by Nisan and Ronen [35] in their seminal paper, the strategic setting for the scheduling problem was not studied before our paper. As we mentioned earlier, the answer to the highlighted question above here is negative, but the setting proved out to be quite rich in terms of the achievable trade-off between the two different inefficiency notions.

To the best of our knowledge, ours is the first paper that proposes the systematic study of the trade-off between the Price of Anarchy and the Price of Stability. While we were preparing our manuscript, we became aware that a trade-off between the two notions was very recently considered also by Ramaswamy et al. [37], though in a fundamentally different setting: they study a special case of covering games, originally introduced by Gairing [18], which is not inherently a mechanism design setup. One the other hand, our interest is in explicitly studying this trade-off in the area of algorithmic mechanism design, thus choosing the prototypical scheduling problem as the starting point.

2 Model and Notation

Let $\mathbb{R}_{\geq 0} = [0, \infty)$ denote the nonnegative reals and $\mathbb{N} = \{1, 2, \dots\}$ the positive integers. For any $n \in \mathbb{N}$, let $[n] = \{1, 2, \dots, n\}$. In the *strategic scheduling* problem (on unrelated machines), there is a set $N = \{1, \dots, n\}$ of *machines* and a set $J = \{1, \dots, m\}$ of *tasks*. Each machine i has a *processing time* (or *cost*) $t_{i,j} \geq 0$ for task j. The induced matrix $\mathbf{t} \in \mathbb{R}_{\geq 0}^{n \times m}$ is the *profile* of processing times. For convenience, we will denote by $\mathbf{t}_i = (t_{i,1}, \dots, t_{i,m})$ the vector of processing times of machine i for the tasks and by $\mathbf{t}^j = (t_{1,j}, \dots, t_{n,j})^{\mathrm{T}}$ the vector of processing times of the machines for task j, so that $\mathbf{t} = (\mathbf{t}_1, \dots, \mathbf{t}_n) = (\mathbf{t}^1, \dots, \mathbf{t}^m)^{\mathrm{T}}$. The machines are *strategic* and therefore, when asked, they do not necessarily report their true processing times $\mathbf{t}$ but they rather use *strategies* $\mathbf{s} \in \mathbb{R}_{\geq 0}^{n \times m}$. To emphasize the distinction, we will often refer to $\mathbf{t}$ as the profile of *true* processing times. Adopting standard game-theoretic notation, we use $\mathbf{t}_{-i}$ and $\mathbf{s}_{-i}$ to denote the profile of true or reported processing times respectively, without the coordinates of the i'th machine.

A (deterministic, direct revelation) *mechanism* $\mathcal{M} = (\mathbf{x}, \mathbf{p})$ gets as input a strategy profile $\mathbf{s} \in \mathbb{R}^{n \times m}$ reported by the machines and outputs *allocation* $\mathbf{x} = \mathbf{x}(\mathbf{s}) \in \{0, 1\}^{n \times m}$ and *payment* $\mathbf{p} = \mathbf{p}(\mathbf{s}) \in \mathbb{R}_{\geq 0}^{n}$: $x_{i,j}$ is an indicator variable denoting whether or not task j is allocated to machine j, and p_i is the payment with which $\mathcal{M}$ compensates machine i for taking part in the mechanism. Thus, the allocation rule needs to satisfy $\sum_{i \in N} x_{i,j}(\mathbf{s}) \geq 1$ for all tasks j.

The *utility* of machine i under a mechanism $\mathcal{M} = (\mathbf{x}, \mathbf{p})$, given true running times $\mathbf{t}_i$ and a reported profile $\mathbf{s}$ by the players, is

$$u_i^{\mathcal{M}}(\mathbf{s}|\mathbf{t}_i) = p_i(\mathbf{s}) - \sum_{j=1}^{m} x_{i,j}(\mathbf{s}) t_{i,j},$$

that is, the payment she receives from $\mathcal{M}$ minus the total workload she has to execute. This is exactly the reason why machines may lie about their true processing times; they will change their report s_i and deviate to another $\mathbf{s}_i'$ if this improves the above quantity. A stable solution with respect to such best-response selfish behaviour is captured by the well-known notion of an equilibrium. Given a mechanism $\mathcal{M}$ and a strategy profile $\mathbf{s}$, we will say that $\mathbf{s}$ is a *(pure Nash) equilibrium*[3] of $\mathcal{M}$ (with respect to a true profile $\mathbf{t}$) if, for every machine i and every possible deviation $\mathbf{s}'_i \in \mathbb{R}^m_{\geq 0}$,

$$u_i^{\mathcal{M}}(\mathbf{s}|\mathbf{t}) \geq u_i^{\mathcal{M}}(\mathbf{s}'_i, \mathbf{s}_{-i}|\mathbf{t}).$$

Notice here that, in general, pure Nash equilibria do not always exist. In this paper we will study only mechanisms that always do have such equilibria. Formally, if $\mathcal{Q}_{\mathbf{t}}^{\mathcal{M}}$ denotes the set of pure Nash equilibria of mechanism $\mathcal{M}$ with respect to true profile $\mathbf{t}$, we will assume that $\mathcal{Q}_{\mathbf{t}}^{\mathcal{M}} \neq \emptyset$ for all $\mathbf{t} \in \mathbb{R}^{n\times m}_{\geq 0}$.

Our objective in this paper is to design mechanisms that minimize the *makespan*

$$C^{\mathcal{M}}(\mathbf{s}|\mathbf{t}) = \max_{i \in N} \sum_{j=1}^{m} x_{i,j}(\mathbf{s}) t_{i,j},$$

that is, the total completion time if our machines run in parallel. For a matrix $\mathbf{t}$ of running times, let $\mathrm{OPT}(\mathbf{t})$ denote the optimum makespan, i.e. $\mathrm{OPT}(\mathbf{t}) = \min_{\mathbf{y}} \sum_{j=1}^{m} y_{i,j} t_{i,j}$ where $\mathbf{y}$ ranges over all feasible allocation of tasks to machines. It is a well-known phenomenon that equilibria can result in suboptimal solutions, and the following, extensively studied, notions where introduced to quantify exactly this discrepancy: the *Price of Anarchy* (PoA) and the *Price of Stability* (PoS) of a scheduling mechanism $\mathcal{M}$ on n machines are, respectively,

$$\mathrm{PoA}(\mathcal{M}) = \sup_{m\in\mathbb{N}, \mathbf{t}\in\mathbb{R}^{n\times m}_{\geq 0}} \frac{\sup_{\mathbf{s}\in\mathcal{Q}_{\mathbf{t}}^{\mathcal{M}}} C^{\mathcal{M}}(\mathbf{s}|\mathbf{t})}{\mathrm{OPT}(\mathbf{t})} \qquad \mathrm{PoS}(\mathcal{M}) = \sup_{m\in\mathbb{N}, \mathbf{t}\in\mathbb{R}^{n\times m}_{\geq 0}} \frac{\inf_{\mathbf{s}\in\mathcal{Q}_{\mathbf{t}}^{\mathcal{M}}} C^{\mathcal{M}}(\mathbf{s}|\mathbf{t})}{\mathrm{OPT}(\mathbf{t})}.$$

For simplicity, we will sometimes drop the $\mathcal{M}$, t and $\mathbf{s}$ in the notation introduced in this section, whenever it is clear which mechanism and which true or reported profile we are referring to.

2.1 Task-Independent Mechanisms

For a significant part of this paper, we will focus on the class of anonymous, task-independent mechanisms. This is a rather natural class of mechanisms; as a matter of fact, two of the arguably most well-studied and used mechanisms in practice, namely the First-Price and Second-Price, lie within this class.

Definition 1 (Task-independence)**.** A mechanism $\mathcal{M} = (\mathbf{x}, \mathbf{p})$ is *task-independent* if each one of its tasks is allocated independently of the others. Formally, there exists a collection of single-task mechanisms $\{\mathcal{A}_j\}_{j=1,\dots,m}$, $\mathcal{A}_j = (\mathbf{y}^j, \mathbf{q}^j)$, such that, for any task j and any machine i,

$$\mathbf{x}^j(\mathbf{s}) = \mathbf{y}^j(\mathbf{s}^j) \qquad \text{and} \qquad p_i(\mathbf{s}) = \sum_{j=1}^{m} q_i^j(\mathbf{s}^j).$$

We will refer to the single-task mechanisms $\mathcal{A}_j$ of the above definition as the *components* of $\mathcal{M}$. It is important to notice here that the definition does not require the mechanism to necessarily use the same component for all the tasks.

Another standard property in the literature of the problem is anonymity. The property can be defined generally (e.g., see [23]), but here we will define it for task-independent mechanisms.

[3] We will be interested in pure Nash equilibria in this paper, but we discuss different solution concepts in Section 2.2 as well as in Section 5.3.

Definition 2 (Anonymity)**.** A single-task mechanism $\mathcal{A} = (\mathbf{x}, \mathbf{p})$ is *anonymous* if, for any inputs $\mathbf{s}, \tilde{\mathbf{s}}$ such that $\tilde{\mathbf{s}} = \pi(\mathbf{s})$ for some permutation[4] π, it holds that

$$\mathbf{x}(\mathbf{s}) = \pi(\mathbf{x}(\tilde{\mathbf{s}})) \qquad \text{and} \qquad \mathbf{p}(\mathbf{s}) = \pi(\mathbf{p}(\tilde{\mathbf{s}})).$$

A task-independent mechanism $\mathcal{M}$ is anonymous, if all its components are anonymous (single-task) mechanisms.

Perhaps the simplest and most natural mechanism that one can think of is the following, which assigns the task to the fastest machine (according to the declared processing times) and pays her her declaration.

Definition 3 (First-Price (FP) mechanism)**.** Assign each task j to the fastest machine $\iota(j)$ for it, i.e. $\iota(j) \in \arg\min_{i \in N} s_{i,j}$ (breaking ties arbitrarily), paying her her declared running time $s_{\iota(i),j}$; pay the remaining $N \setminus \{\iota(j)\}$ machines 0 for task j.

Second-Price mechanisms have also been extensively studied and applied in auction theory, but also in strategic scheduling. As we mentioned in the introduction, the following mechanism is usually referred to as the VCG mechanism in the literature of the problem (see e.g.,[7]):

Definition 4 (Second-Price (SP) mechanism)**.** Assign each task j to the fastest machine $\iota(j)$ for it, i.e. $\iota(j) \in \arg\min_{i \in N} s_{i,j}$ (breaking ties arbitrarily), paying her the declared processing time of the second-fastest machine, i.e. $\min_{i \in N \setminus \{\iota(j)\}} s_{i,j}$; pay the remaining $N \setminus \{\iota(j)\}$ machines 0 for task j.

Notice that both FP and SP mechanisms are task-independent and anonymous. Furthermore, SP is truthful. As a matter of fact, SP is the best known truthful mechanism if one is interested only in dominant strategy equilibria (see, e.g., [35, 9] and Section 2.2).

2.2 Solution Concepts and Notions of Inefficiency

The solution concept that we consider in this paper is that of the pure Nash equilibrium. In the literature of the truthful scheduling problem, the employed solution concept is that of the *dominant strategy equilibrium*, i.e., a strategy profile in which no agent would not have an incentive to deviate to any other strategy, for any possible strategy vector of the remaining agents. More precisely, the literature has been interested in *truthful* mechanisms, i.e., mechanisms for which truth-telling is always (i.e., for any processing time profile $\mathbf{t}$) a dominant strategy equilibrium. The goal is to find a mechanism with the best *approximation ratio*, which is defined as the worst-case (over all inputs) ratio of the makespan of the mechanism over the optimal makespan, in the truth-telling equilibrium.

For this objective, studying only the truth-telling dominant strategy equilibria is without loss of generality, by the Revelation Principle (see, e.g., [34]). We remark however that the fact that an allocation function can be implemented in truth-telling dominant strategies (i.e., an appropriate payment function can be found such that the resulting mechanism is truthful) does not have any direct implications on the space of non-truthful mechanisms and their PoA/PoS guarantees.

There are however some inherent relations between the approximation ratio, the Price of Anarchy and the Price of Stability which follow directly from their definitions. Clearly, a Price of Anarchy guarantee is stronger than a Price of Stability guarantee, since the former bounds the inefficiency of all equilibria while the latter is only concerned with the best one. Since

[4] For any permutation $\pi : \{1, \dots, n\} \longrightarrow \{1, \dots, n\}$ and n-dimensional vector $\mathbf{x} = (x_1, \dots, x_n)$, the permutation of $\mathbf{x}$ under π is the vector $\pi(\mathbf{x}) \equiv (x_{\pi(1)}, \dots, x_{\pi(n)})$.

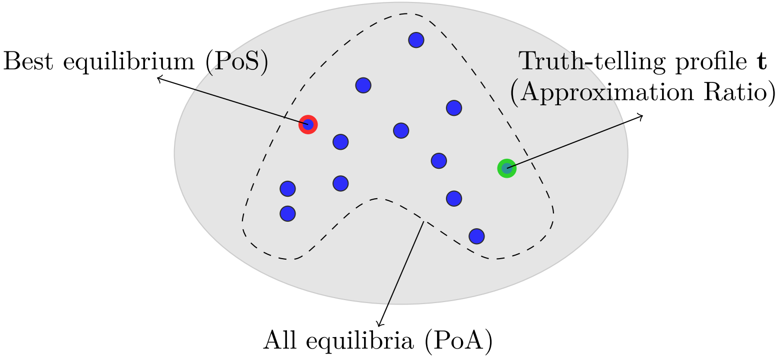

Figure 1: A pictorial representation of the relation between the different solution concepts and the notions of inefficiency. The picture shows the set of equilibria of a mechanism $\mathcal{M}$ for a given true profile of processing times $\mathbf{t}$ (they are depicted as a finite set, for convenience, but this need not be the case). The gray area is the set of all strategy profiles $\mathbf{s}$ and the blue nodes are the pure Nash equilibria of $\mathcal{M}$ on $\mathbf{t}$. The best equilibrium on this profile is drawn with a red boundary (we assume that there is a single best equilibrium for easy of exposition) and the truth-telling profile is drawn with a green boundary. Note that if $\mathcal{M}$ is truthful, then the truth-telling profile is a (dominant strategy) equilibrium. The PoA bounds the inefficiency (over all possible profiles $\mathbf{t}$) for all the blue nodes, the PoS bounds the inefficiency for the node with the red boundary and the approximation ratio bounds the inefficiency for the node with the green boundary.

dominant strategy equilibria are also Nash equilibria by definition, for truthful mechanisms, a Price of Anarchy guarantee is also stronger than an approximation ratio guarantee, which, in turn, is stronger than a Price of Stability guarantee. An illustration of the relation between these different notions is given in Fig. 1.

3 The Inefficiency of *All* Mechanisms

We start with a lower bound of n for the Price of Anarchy of the scheduling problem, which applies to *all* mechanisms. The lower bound will be based on the following monotonicity lemma. We note that this monotonicity property is different from the weak monotonicity (WMON) used in the literature of truthful machine scheduling (see e.g. [35, 9]) in the sense that (a) it is global, whereas WMON is local and (b) it applies to the relation between the true processing times and the equilibria of the mechanism, rather than the actual allocations.

Lemma 1 (Equilibrium Monotonicity)**.** *Let $\mathcal{M}$ be any mechanism for the scheduling problem. Let $\mathbf{t}$ be a profile of true processing times and let $\mathbf{s} \in \mathcal{Q}_{\mathbf{t}}$ be an equilibrium under $\mathbf{t}$. Denote by S_i the set of tasks assigned to machine i by $\mathcal{M}$ on input $\mathbf{s}$. Consider any profile $\hat{\mathbf{t}}$ such that for every machine i, $\hat{t}_{i,j} \leq t_{i,j}$ if $j \in S_i$ and $\hat{t}_{i,j} \geq t_{i,j}$ if $j \notin S_i$. Then $\mathbf{s} \in \mathcal{Q}_{\hat{\mathbf{t}}}$, i.e., $\mathbf{s}$ is an equilibrium under $\hat{\mathbf{t}}$ as well.*

Proof. Assume by contradiction that $\mathbf{s} \notin \mathcal{Q}_{\hat{\mathbf{t}}}$, which means that for the profile of processing times $\hat{\mathbf{t}}$, there exists some machine i that has a beneficial deviation $\mathbf{s}'_i$, i.e. $u_i(\mathbf{s}'_i, \mathbf{s}_{-i}|\hat{\mathbf{t}}) > u_i(\mathbf{s}|\hat{\mathbf{t}})$. Let S'_i be the set of tasks assigned to machine i under report $\mathbf{s}' = (\mathbf{s}'_i, \mathbf{s}_{-i})$ (and underlying true reports $\hat{\mathbf{t}}$). The difference in utility for machine i between profiles $\mathbf{s}'$ and $\mathbf{s}$ is

$$\Delta u_i(\hat{\mathbf{t}}) \equiv u_i(\mathbf{s}'|\hat{\mathbf{t}}) - u_i(\mathbf{s}|\hat{\mathbf{t}}) = p_i(\mathbf{s}') - p_i(\mathbf{s}) + \sum_{j \in S_i \setminus S'_i} \hat{t}_{i,j} - \sum_{j \in S'_i \setminus S_i} \hat{t}_{i,j}.$$

By the fact that s'_i is a beneficial deviation, it holds that $\Delta u_i(\hat{\mathbf{t}}) > 0$.

Now consider the profile of processing times $\mathbf{t}$ and the same deviation $\mathbf{s}'_i$ of machine i. The increase in utility now is

$$\Delta u_i(\mathbf{t}) = p_i(\mathbf{s}') - p_i(\mathbf{s}) + \sum_{j \in S_i \setminus S'_i} t_{i,j} - \sum_{j \in S'_i \setminus S_i} t_{i,j} \geq p_i(\mathbf{s}') - p_i(\mathbf{s}) + \sum_{j \in S_i \setminus S'_i} \hat{t}_{i,j} - \sum_{j \in S'_i \setminus S_i} \hat{t}_{i,j} = \Delta u_i(\hat{\mathbf{t}}),$$

which holds because $t_{i,j} \geq \hat{t}_{i,j}$, if $j \in S_i$ and $t_{i,j} \leq \hat{t}_{i,j}$, if $j \notin S_i$. This implies that $\Delta u_i(\mathbf{t}) > 0$, which contradicts the fact that $\mathbf{s} \in \mathcal{Q}_{\mathbf{t}}$. □

Using this lemma, we can prove our first lower bound:

Theorem 1. *For any scheduling mechanism $\mathcal{M}$ for n machines, it must be that* $\mathrm{PoA}(\mathcal{M}) \geq n$.

Proof. Let $\mathcal{M}$ be any mechanism and consider a profile of true processing times $\mathbf{t}$ with n machines and n^2 tasks, where $t_{i,j} = 1$ for all machines i and all tasks j. Let $\mathbf{s} = (s_1, s_2, \dots, s_n)$ be a pure Nash equilibrium of $\mathcal{M}$ under $\mathbf{t}$. For each machine i, let S_i be the set of tasks assigned to that machine and note that there exists some machine k for which $|S_k| \geq n$. Let $T_k \subseteq S_k$ be any subset of S_k such that $|T_k| = n$.

Now consider the following profile $\hat{t}$ of processing times:

- For all $i \neq k$, $\hat{t}_{i,j} = 0$, for all $j \in S_i$ and $\hat{t}_{i,j} = t_{i,j}$, for all $j \notin S_i$.
- $\hat{t}_{kj} = 0$, for all $j \in S_k \setminus T_k$ and $\hat{t}_{kj} = t_{k,j}$, for all $j \notin S_k \setminus T_k$.

By Lemma 1, the profile $\mathbf{s} = (s_1, s_2, \dots, s_n)$ is a pure Nash equilibrium under $\hat{\mathbf{t}}$ and the allocation is the same as before, for a makespan of at least n, since machine k is assigned all the tasks in T_k. The optimal allocation will assign one task from T_k to each machine, the tasks from S_i to machine i for each $i \neq k$ and the tasks from $S_k \setminus T_k$ to machine k, for a total makespan of 1 and the Price of Anarchy bound follows. □

3.1 PoA/PoS Trade-off

In this section, we prove our main theorem regarding the trade-off between the Price of Anarchy and the Price of Stability. The theorem informally says that if the Price of Stability of a mechanism is small, then its Price of Anarchy has to be high.

Theorem 2. *For any scheduling mechanism $\mathcal{M}$ for $n \geq 2$ machines, and any real $\rho > 1$,*

$$\mathrm{PoS}(\mathcal{M}) < \rho \quad \Longrightarrow \quad \mathrm{PoA}(\mathcal{M}) \geq \frac{n-1}{\rho - 1}$$

Proof. Consider an instance with n players and n tasks. Assume a true $n \times n$ processing-times matrix $\mathbf{t}$ with

$$t_{1,j} = \begin{cases} n-1, & \text{if } j = 1, \\ \rho - 1, & \text{otherwise,} \end{cases}$$

and

$$t_{i,j} = \begin{cases} n-1, & \text{if } j = i, \\ \infty, & \text{otherwise,} \end{cases}$$

for all $i = 2, \dots, n$. Here ∞ denotes an arbitrarily large positive value, and actually replacing it with any value $M \geq \rho(n-1)$ will work just fine for our proof.[5]

[5] We will adopt a similar convention throughout the paper.

First notice that by allocating each task j to machine j with running time $t_{j,j} = n - 1$, for all $j \in [n]$, we get an upper bound of $n - 1$ on the optimal makespan of $\mathbf{t}$. Thus, since $\text{PoS}(\mathcal{M}) < \rho$, there must exist a pure Nash equilibrium profile $\mathbf{s}^\star$ such that the allocation $\mathcal{M}(\mathbf{s}^\star)$ results in a makespan less than $\rho(n - 1)$ (with respect to the underlying, true time matrix $\mathbf{t}$). But then, due to the structure of $\mathbf{t}$, and in particular the large value of M, $\mathcal{M}(\mathbf{s}^\star)$ can only allocate each task j to either machine 1 or machine j, for all $j \in [n]$. In particular, task 1 will necessarily have to be allocated to machine 1. Furthermore, from the remaining $n - 1$ tasks, not all of them can be allocated to machine 1, because that would give rise to a running time of $t_{1,1} + \sum_{j=2}^{n} t_{1,j} = n - 1 + (n - 1)(\rho - 1) = \rho(n - 1)$ for machine 1, which violates the Price of Stability constraint assumed for $\mathbf{s}^\star$. So, there must exist at least one task $j \geq 2$, denote it by $j^\star$, such that $\mathcal{M}(\mathbf{s}^\star)$ allocates j to machine j.

For each task j, let i_j denote the machine which task j is allocated to by $\mathcal{M}(\mathbf{s}^\star)$. Now modify the original, true execution time matrix $\mathbf{t}$ by changing the running time $t_{i_j,j}$, for all $j \neq j^\star$, to $t'_{i_j,j} = 0$. Denote this new matrix by $\mathbf{t}'$. Due to Lemma 1, $\mathbf{s}^\star$ has to be a pure Nash equilibrium of $\mathcal{M}$ with respect to the modified true profile $\mathbf{t}'$ as well. But now $\mathcal{M}(\mathbf{s}^\star)$ results in a makespan of at least $t'_{j^\star,j^\star} = t_{j^\star,j^\star} = n - 1$ (since task $j^\star$ is allocated to machine $j^\star$), while allocating $j^\star$ to machine 1 (and leaving all other assignments as they are, i.e. task $j \neq j^\star$ gets allocated to machine i_j) results in machine 1 having a total running cost of at most $(n-1) \cdot 0 + t'_{1,j^\star} = \rho - 1$, and all other machines 0. This gives a Price of Anarchy lower bound of $\frac{n-1}{\rho-1}$. □

By allowing ρ in Theorem 2 to go arbitrarily close to 1 we get the following:

Corollary 1. *Even for just two machines, if a scheduling mechanism has an optimal Price of Stability of* 1*, then its Price of Anarchy has to be unboundedly large.*

From the results of the section, we well as the trivial fact that $\text{PoA}(\mathcal{M}) \geq \text{PoS}(\mathcal{M})$ for any mechanism $\mathcal{M}$, we obtain a feasibility trade-off between the PoA and the PoS of scheduling mechanisms, which is illustrated in Fig. 2. We refer to the boundary of the shaded feasible region as the *inefficiency boundary*; the shape of the boundary follows from Theorem 2, as well as Theorem 1, since for any $\rho > (n-1)/n + 1$, the largest lower bound on the Price of Anarchy is given by Theorem 1.

3.1.1 Mechanisms on the Extrema of the Inefficiency Boundary

When looking for mechanisms on the Pareto frontier, the first ones that come to mind are perhaps the First-Price (FP) and Second-Price (SP) mechanisms, defined in Section 2, which are straightforward adaptations of the well-known First-Price auction and Second-Price auction mechanisms from the auction literature.

It follows from known results in the literature for the First-Price auction (see, e.g., [16]) that in every pure Nash equilibrium of the FP, each task is allocated to the machine with the smallest *true* processing time for the task. In Section 4.2, we will define a class of task-independent mechanisms ($\mathcal{SP}_\alpha$) that contain FP as a corner case ($\mathcal{SP}_1$) and in particular, the following result can also be obtained as corollary of our more general results in Theorem 6, Theorem 7 and Theorem 8, when applied for $\alpha = 1$.

Theorem 3. *For the First-Price mechanism, it holds that* $\text{PoA}(\text{FP}) = \text{PoS}(\text{FP}) = n$.

For the Second-Price mechanism, again it follows from known observations in the literature of auctions that while the mechanism is truthful, it has several other pure Nash equilibria as well. More precisely, for a task $j \in J$ and any machine $i \in N$, there exists an equilibrium for which task j is allocated to machine i. This immediately implies the following.

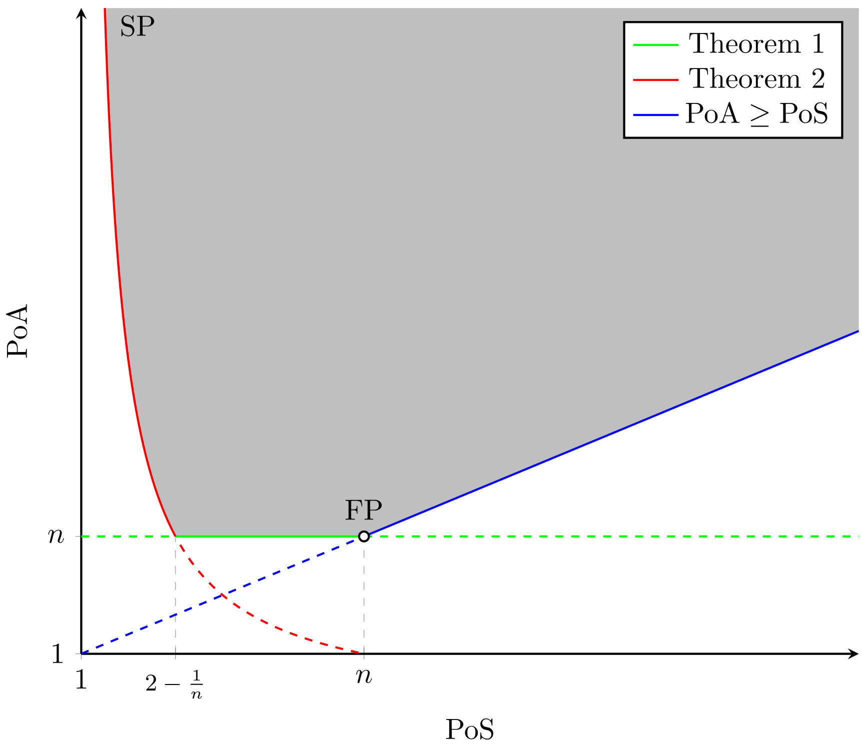

Figure 2: The inefficiency boundary for general mechanisms, given by Theorem 2 (red line). Combined with global PoA lower bound of Theorem 1 (green line) and the trivial fact that the PoS is at most PoA (blue line), we finally get the grey feasible region.

Theorem 4. *For the Second-Price mechanism, the* PoA *of the mechanism is unbounded and* $\text{PoS}(\text{SP}) = 1$.

Again, Theorem 4 can be obtained as a corollary of our results in Section 4.2, since SP is also a corner-case mechanism in our class, namely $\mathcal{SP}_\infty$. Interestingly, we identify that the bad PoA bound is inherently present in the constraints that truthfulness enforces on the payment functions. The following theorem highlights that if one is interested in the set of *all* equilibria, they would have to reach out beyond truthful mechanisms.

Theorem 5. *The Price of Anarchy of any truthful mechanism is unbounded.*

Proof. Let $\mathcal{M} = (\mathbf{x}, \mathbf{p})$ be a truthful mechanism on $n \geq 2$ machines. To arrive to a contradiction, assume that there exists a real $M \geq 1$ such that $\text{PoA}(\mathcal{M}) \leq M$. We will consider single-task instances with only $n = 2$ machines. This is without loss of generality, since one can add arbitrarily many more machines with 0 running time for the task, and the proof remains valid. In particular, we assume an underlying vector of true running times $\mathbf{t} = (0, \varepsilon)$, where $\varepsilon \in (0, M^{-1})$, and a vector $\mathbf{s} = (1, 0)$ of reported costs. We first show that $\mathcal{M}$ allocates the task to machine $i = 2$, even if she reports slightly slower running times.

Claim 1. For any $\delta \in [0, \varepsilon]$, mechanism $\mathcal{M}$ always allocates the task to machine $i = 2$ on any input $\mathbf{s}' = (1, \delta)$.

Proof. First notice that, due to truthfulness, if we consider as true underlying profile $\mathbf{t}' = \mathbf{s}' = (1, \delta)$, then $\mathbf{s}'$ has to be an equilibrium. Next, for a contradiction, assume that there exists a nonnegative $\delta \leq \varepsilon$ such that $x_2(\mathbf{s}') = 0$. Then $x_1(\mathbf{s}') = 1$, and thus the makespan under equilibrium $\mathbf{s}'$ (and true profile $\mathbf{t}'$) would be 1, while an optimal solution would have given the task to the faster machine, for a makespan of δ. This results in a PoA of at least $\frac{1}{\delta} \geq \varepsilon^{-1} > M$ which is a contradiction. This completes the proof of the claim. ■

Using the same argument, we can also show that $\mathcal{M}$ keeps allocating the task to machine $i = 2$ as long as the other machine has a strictly positive cost:

Claim 2. For any $\delta > 0$, mechanism $\mathcal{M}$ always allocates the task to machine $i = 2$ on any input $\mathbf{s}' = (\delta, 0)$.

We are now ready to prove that $\mathbf{s}$ is actually an equilibrium:

Claim 3. Reporting $\mathbf{s} = (1, 0)$ is an equilibrium of $\mathcal{M}$ (under true costs $\mathbf{t}$).

The above claim is enough to complete the proof, since by combining it with the previous Claim 1 (with $\delta = 0$) we get that the makespan of $\mathcal{M}$ under equilibrium $\mathbf{s}$ is $\varepsilon > 0$ (since the task goes to machine $i = 2$) while the optimal one is 0. This contradicts the fact that PoA($\mathcal{M}$) is bounded.

Proof of Claim 3. First we remark that, due to well-known characterizations of truthfulness for single-dimensional domains [32] (which apply to our case, since we have a single task), there exist real functions h_1, h_2 such that the utilities of our machines (with respect to true costs $\mathbf{t}$) on any vector of reports $\mathbf{s}'$ are given by[6]

$$u_i(\mathbf{s}') = h_i(s'_{-i}) + (s'_i - t_i) \cdot x_i(\mathbf{s}') - \int_0^{s'_i} x_i(z, s'_{-i})\, dz. \tag{1}$$

Furthermore, the allocation function $x_i(\mathbf{s}')$ of each player i is monotonically nonincreasing with respect to her reported cost s'_i.

Now we show that the first player has no incentive to deviate from reporting $s_1 = 1$ as long as the second player is fixed at $s_2 = 0$. Indeed, currently, and as long as she reports any strictly positive cost $s'_1 = z > 0$, she will still lose the task (i.e., $x_1(z, s_2) = 0$). That holds due to Claim 2. From (1), this results in a utility of $u_1(s'_1, s_2) = h_1(s_2)$. For the only remaining case that she reports $s_1 = 0'$, again from (1) we get that $u_1(0, s_2) = h_1(s_2) - (0-0) \cdot x_1(0, s_2) = h_1(s_2)$. Thus, in no case the first player can gain by unilaterally deviating.

Finally, we need to show that the second player has no incentive to deviate from reporting $s_2 = 0$ as well. Assuming the other player fixed at $s_1 = 1$, the improvement in her utility by declaring a cost $s'_2 \geq 0$ is (due to (1))

$$u_2(1, s'_2) - u_2(1, 0) = (s'_2 - \varepsilon) x_2(1, s'_2) - \int_0^{s'_2} x_2(1, z)\, dz.$$

Recall now from Claim 1 that $x_1(1, z) = 1$ for all $z \in [0, \varepsilon]$. Thus, for $s'_2 \in [0, \varepsilon]$ the above difference in the second player's utility becomes $(s'_2 - \varepsilon) - s'_2 = -\varepsilon < 0$, while for $s'_2 > \varepsilon$ it is

$$(s'_2 - \varepsilon) x_2(1, s'_2) - \int_\varepsilon^{s'_2} x_2(1, z)\, dz \leq (s'_2 - \varepsilon) x_2(1, s'_2) - (s'_2 - \varepsilon) x_2(1, s'_2) x_2(1, s'_2) = 0,$$

the inequality holding due to the fact that $x_2(1, z)$ is nonincreasing with respect to z. ■

□

From Theorem 2, Theorem 3 and Theorem 4, it is clear that both FP and SP lie on the boundary of the PoA/PoS feasibility space, see Fig. 2.

[6]See [2, Theorem 4.2].

4 The Pareto Frontier of Task-Independent Mechanisms

As we noted in the previous section, both the SP and FP mechanisms, which lie on the inefficiency boundary (see Fig. 2), are anonymous task-independent mechanisms. In this section, we will construct a tighter boundary on the PoA/PoS trade-off for the class of anonymous task-independent mechanisms. Furthermore, we will show that this boundary is actually tight, by designing a class of optimal mechanisms that lie exactly on it, meaning that for each point on the boundary, there is a mechanism in our class that achieves the corresponding PoA/PoS trade-off. Thus, this results in a *complete characterization of the Pareto frontier* between PoA and PoS.[7] For an illustration, see Fig. 3.

4.1 PoA/PoS Trade-off

We start with the theorem that gives us the improved boundary on the space of feasible task-independent and anonymous mechanisms. This is the red line in Fig. 3.

Theorem 6. *For any task-independent anonymous scheduling mechanism $\mathcal{M}$ for n machines, and any real $\alpha > 1$,*

$$\mathrm{PoA}(\mathcal{M}) < (n-1)\alpha + 1 \quad \Longrightarrow \quad \mathrm{PoS}(\mathcal{M}) \geq \frac{(n-1)}{\alpha} + 1.$$

Proof. Fix a mechanism $\mathcal{M}$ that allocates each task j independently, by running an anonymous single-task mechanism $\mathcal{A}_j$. Each such mechanism $\mathcal{A}_j$ takes as input a declared cost vector $\mathbf{s}^j = (s_{1,j}, \dots, s_{n,j})$ by the machines, where s_i is the report of machine i for task j, $i = 1, \dots, n$. Of course, there is also an underlying *true* cost vector $\mathbf{t}^j = (t_{1,j}, \dots, t_{n,j})$. Also, fix a parameter $\alpha > 1$.

We are particularly interested in true cost vectors that have a specific structure, namely being permutations of $(1, \alpha, \infty, \dots, \infty)$. We will call such cost vectors *canonical* for the remainder of this proof. Equivalently, $\mathbf{t}^j$ is canonical if:

- there is a unique machine i' such that $t_{i',j} = 1$ (machine i' will be called *fast* for task j),
- there is a unique machine i'' such that $t_{i'',j} = \alpha$ (machine i'' will be called *slow* for task j),
- all other machines $i \neq i', i''$ have arbitrarily high processing time $t_{i,j} = \infty$ for task j.

It is important to notice here that, due to anonymity, if $\mathbf{s}^j$ is a pure Nash equilibrium of mechanism $\mathcal{A}_j$ under a canonical true cost vector $\mathbf{t}^j$, on which $\mathcal{A}_j$ allocates the task to the slow machine, then under *any* other canonical true vector $\hat{\mathbf{t}}^j$ there has to exist an equilibrium $\hat{\mathbf{s}}^j$ on which the task is again allocated to the slow machine. More concretely, let π be the permutation such that $\hat{\mathbf{t}}^j = \pi(\mathbf{t}^j)$. Then, because $\mathcal{A}_j$ is anonymous, its allocation on input $\hat{\mathbf{s}}^j = \pi(\mathbf{s}^j)$ is also permuted according to π, thus preserving the assignment of the task to the machines with the same running time between the old and new true vectors. Also, it is easy to see that the exactly analogous argument holds for the allocation to fast machines.

[7] To prevent any potential confusion, we use the term "inefficiency boundary" to refer to the boundary of the feasible region for the PoA/PoS trade-off, that is defined by some impossibility-type result such as Theorem 6 and we reserve the term "Pareto frontier" for a boundary that can provably not be improved, since there are mechanisms that achieve the corresponding trade-offs. Intuitively, in our terminology, the inefficiency boundary is a "bound" on the achievable Pareto frontier.

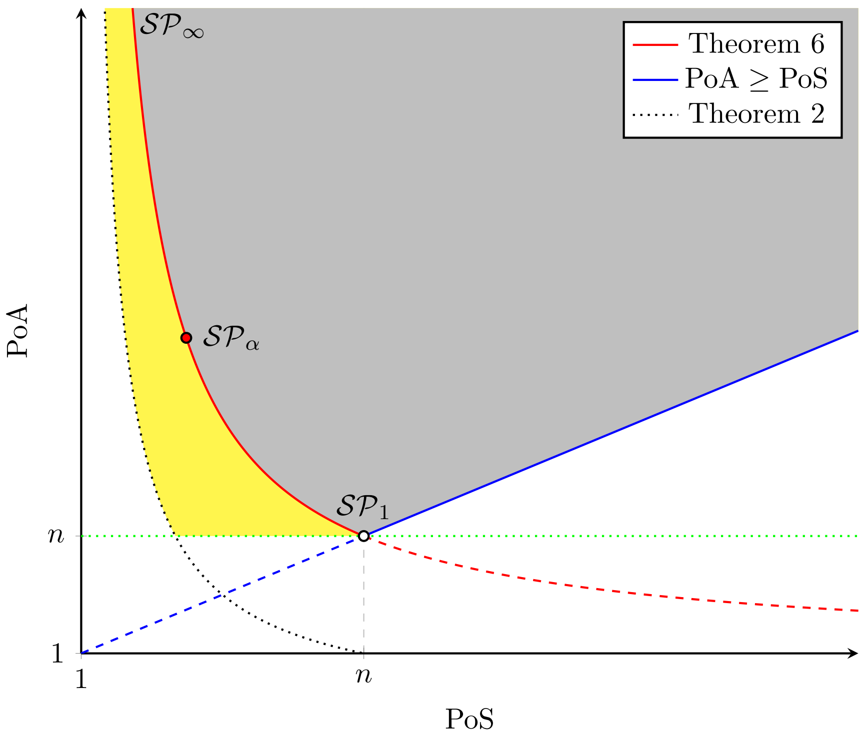

Figure 3: The inefficiency boundary, for anonymous task-independent mechanisms, given by Theorem 6 (red line). Combined with global PoA lower bound of Theorem 1 (green line) and the trivial fact that the PoS is at most PoA (blue line), we finally get the grey feasible region. The family of mechanisms $\mathcal{SP}_\alpha$ described in Section 4.2 lies exactly on this boundary (red line), thus completely characterizing the *Pareto frontier* in a smooth way with respect to parameter $\alpha \geq 1$: on its one end ($\alpha = 1$) is the First-Price mechanism FP $= \mathcal{SP}_1$ and at the other ($\alpha \to \infty$) the Second-Price mechanism SP $= \mathcal{SP}_\infty$.

Now consider a true instance $\mathbf{t}$ consisting of $m \geq 2n-3$ tasks that all have canonical cost vectors. Further, assume that the PoA of $\mathcal{M}$ under $\mathbf{t}$ is less than $(n-1)\alpha+1$. We will first argue that, at *any* equilibrium (under true costs $\mathbf{t}$), $\mathcal{M}$ has to allocate at least $n-1$ tasks to fast machines. To get to a contradiction, assume that there exists an equilibrium $\mathbf{s}$ under which $\mathcal{M}$ allocates at most $n-2$ tasks to their fast machines. Then, at least $m-(n-2) \geq n-1$ tasks have to go to their slow machines. Without loss, for simplicity let's assume that that is the case for tasks $j = 1, \dots, n-1$. Then, as discussed above, we can permute the cost vectors of these tasks in any way we want, and still be sure that there exists an equilibrium under which all these tasks go to their slow machines.

In particular, consider the following new profile of true costs

$$\hat{\mathbf{t}} = \begin{pmatrix} 1 & \infty & \infty & \cdots & \infty \\ \infty & 1 & \infty & \cdots & \infty \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ \infty & \cdots & \infty & 1 & \infty \\ \alpha & \alpha & \cdots & \alpha & 1 \end{pmatrix},$$

which results from $\mathbf{t}$ by permuting accordingly the canonical task vectors of the first $j = 1, \dots, n-1$ tasks, setting the running times of task $j = n$ to be arbitrarily slow except from machine $i = n$ that can execute for a cost of 1, and finally, setting the running times of all other tasks $j \geq n+1$ equal to 0, thus rendering them essentially irrelevant. Then, there exists an equilibrium $(\hat{\mathbf{s}}_1, \dots, \hat{\mathbf{s}}_{n-1}, \hat{\mathbf{s}}_n)$ that allocates all tasks to machine $i = n$: for all $j \leq n-1$, $\hat{\mathbf{s}}_j$

allocates the task to its slow machine, which happens to be the same for all j, namely $i = n$; for $j = n$, $\hat{\mathbf{s}}_n$ necessarily gives the task to machine $i = n$, since it is the only one with a bounded running time. This results in $\mathcal{M}$ having a makespan of at least $(n-1)\alpha+1$. On the other hand, assigning tasks according to the diagonal, that is, giving each task j to machine $i = j$, would result in a makespan of 1. From the above, we deduce that $\text{PoA}(\mathcal{M}) \geq (n-1)\alpha + 1$, which is a contradiction.

Thus, from now on we assume that indeed all equilibria of $\mathcal{M}$, under our initial true costs $\mathbf{t}$, have to allocate at least $n-1$ tasks to their fast machines. Without loss, assume now that this is the case for the first $j = 1, \dots, n-1$ tasks. Now consider the following modified instance $\hat{\mathbf{t}}$ of true costs, that results from $\mathbf{t}$ by permuting the costs vectors of tasks $j = 1, \dots, n-1$ so that they all have the same fast machine (namely, $i = n$), making task $j = n$ having a finite running time of α only for machine n, and rendering all remaining tasks $j \geq n+1$ irrelevant by setting their running times at 0:

$$\hat{\mathbf{t}} = \begin{pmatrix} \alpha & \infty & \infty & \cdots & \infty \\ \infty & \alpha & \infty & \cdots & \infty \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ \infty & \cdots & \infty & \alpha & \infty \\ 1 & 1 & \cdots & 1 & \alpha \end{pmatrix}$$

Any equilibrium $\hat{\mathbf{s}} = (\hat{\mathbf{s}}_1, \dots, \hat{\mathbf{s}}_{n-1}, \hat{\mathbf{s}}_n)$ of $\mathcal{M}$ under $\hat{\mathbf{t}}$ needs to assign all tasks $j = 1, \dots, n-1$ to the fast machine, that is, to $i = n$: otherwise, as we have argued before, there would exist an equilibrium under the original true profile $\mathbf{t}$ on which $\mathcal{M}$ would allocate one of the tasks $j = 1, \dots, n-1$ to a slow machine, which is a contradiction. Thus, any equilibrium equilibrium $\hat{\mathbf{s}}$ results in $\mathcal{M}$ having a makespan of $(n-1) + \alpha$, while the diagonal allocation on $\hat{\mathbf{t}}$ has a makespan of α. This results in $\text{PoS}(\mathcal{M}) \geq \frac{n-1}{\alpha} + 1$. □

4.2 Optimal Mechanisms on the Pareto Frontier

Next, we will design a class of mechanisms, parameterized by a quantity α that will populate, in a smooth way, the boundary given by Theorem 6. Thus, these mechanisms achieve trade-offs that lie on the Pareto frontier of inefficiency for the class of task-independent and anonymous mechanisms.

Definition 5 (Second-Price mechanism with α-relative reserve price ($\mathcal{SP}_\alpha$))**.** For $\alpha \geq 1$, $\mathcal{SP}_\alpha$ is the task-independent mechanism that, for each task j: finds a machine $k \in \arg\min_{i \in N} s_{i,j}$ and sets a *reserve price* at $r = \alpha \cdot s_{k,j}$; assigns the task to the fastest machine $\iota(j) \in \text{argmin}_{i \in N} s_{i,j}$ (breaking ties-arbitrarily); pays machine $\iota(j)$ the amount $\min\{\min_{i \in N \setminus \{\iota(j)\}} s_{i,j}, r\}$; pays nothing to the remaining machines $N \setminus \iota(j)$.

Informally, for each task j, the mechanism sets a reserve price which is α times larger than the smallest declared processing time, allocates the task to the fastest machine (according to the declarations) and pays the machine the minimum of the second-smallest declared processing time and the reserve price. What this mechanism achieves in terms of the equilibria that it induces is the following: assume that we create a *bucket* of tasks with *true* processing times at most α times larger than the smallest *true* processing time. Then, in every equilibrium of the mechanism, task j is allocated to some machine in the bucket and moreover, for any machine in the bucket, there exists some equilibrium under which $\mathcal{SP}_\alpha$ allocates the task to that machine. This is captured formally by the following two lemmas. Referencing our discussion in Section 3.1.1, we remark that in the case of SP $= \mathcal{SP}_1$, the bucket contains only the fastest machine(s) for the task, and in the case of FP $= \mathcal{SP}_\infty$, the bucket contains the whole set of machines.

Lemma 2 ("Nothing outside the bucket"). *In any equilibrium of $\mathcal{SP}_\alpha$, any task $j \in J$ can only be assigned to a machine with processing time at most $\alpha \cdot \min_{i\in N} t_{i,j}$.*

Proof. Since $\mathcal{SP}_\alpha$ is task-independent, it suffices to consider the equilibria of a single component $\mathcal{SP}_\alpha^j$, corresponding to task j. Let $\mathbf{s}^j$ be such an equilibrium and without loss of generality, assume that $t_{1,j} \in \arg\min_{i\in N} t_{i,j}$, i.e. Machine 1 is the fastest machine according to the true processing times. Assume by contradiction that in $\mathbf{s}^j$, some task ℓ with real processing time $t_{\ell,j} > \alpha \cdot t_{1,j}$ is allocated task j and let $s_{\ell,j}$ be its report. Since Machine ℓ receives the task, it obviously holds that $s_{\ell,j} \in \arg\min_{i\in N} s_{i,j}$. We will consider two cases.

Case 1: $s_{\ell,j} \leq t_{1,j}$. In this case, the reserve price is set at $r = \alpha \cdot s_{\ell,j} \leq \alpha \cdot t_{1,j}$ and Machine ℓ receives a payment of at most $\alpha \cdot t_{1,j}$. By assumption however, its true processing time is larger than $\alpha \cdot t_{i,j}$ and therefore the machine has negative utility. By deviating to telling the truth, the machine can obtain a nonnegative utility, contradicting the fact that $\mathbf{s}^j$ is an equilibrium.

Case 2: $s_{\ell,j} > t_{1,j}$. In this case, Machine 1 has 0 utility, since she is not allocated the task and she is not paid anything. However, if Machine 1 deviates to telling the truth, (i.e., if she deviates to $s'_{i,j} = t_{1,j}$), then, since $s_{\ell,j} = \min i \in N s_{i,j}$ by assumption, the machine will now win the task (i.e., $x_{1j} = 1$) and will receive a payment of $s_{\ell,j} > t_{1,j}$, obtaining strictly positive utility. Again, this contradicts the fact that $\mathbf{s}^j$ is an equilibrium.

In any case, $\mathbf{s}^j$ can not be an equilibrium in which task j is allocated to some machine with processing time larger than $\alpha \cdot \min_{i\in N} t_{i,j}$. □

Lemma 3 ("Everything inside the bucket"). *For every input profile $\mathbf{t}$, there exist equilibria of $\mathcal{SP}_\alpha$ such that for every task j, every machine with processing time at most $\alpha \cdot \min_{i\in N} t_{i,j}$ can be allocated task j.*

Proof. Again, since $\mathcal{SP}_\alpha$ is task-independent, it suffices to consider the equilibria of a single component $\mathcal{SP}_\alpha^j$, corresponding to task j. Given an input profile $\mathbf{t}$, let J_α be the set of machines i' such that $t_{i',j} = \alpha \cdot \min_{i\in N} t_{i,j}$ and let $t_f = \min_{i\in N} t_{i,j}$ be the processing time of the fastest machine for task j. Let k be any machine in J_α and consider the strategy profile $\mathbf{s}$ (or rather, its restriction to the j-th component $\mathbf{s}^j$) such that $s_{k,j} = t_f$, $s_{i,j} = t_{k,j}$ for all machines $i \in J_\alpha \setminus \{k\}$ and $s_{i,j} = t_{i,j}$ for all machines $i \in J \setminus J_\alpha$. We will prove that $\mathbf{s}^j$ is an equilibrium, by considering possible deviations of Machine k, and the remaining machines in $J \setminus \{k\}$ separately.

- The current utility of Machine k is 0, since it receives the task for which the true processing time is $t_{k,j}$ and receives a payment of $t_{k,j}$. Note that the reserve price is set to at most $t_{k,j}$, since $t_{k,j} \leq \alpha \cdot t_f$ by assumption, and $s_{k,j} = t_f$. Since there exist machines with reported processing times at $t_{k,j}$, Machine k can not obtain a positive utility by any deviation, even if it increases the reserve price while still winning the task.

- Consider any machine $i \in J \setminus \{k\}$. Since Machine i is not winning the task, its utility is 0, and the only way to possibly obtain a positive utility is by forcing an allocation in which it wins the task. For this to be possible, it has to deviate to $s'_{i,j} \leq t_f = s_{k,j}$, as otherwise Machine k would still be the winner. In that case however, the payment of Machine i will be at most t_f and by assumption, we know that $t_{i,j} \geq t_f$ for any task $j \in J$. Therefore, the deviation results in a utility of at most 0 for Machine i and it is not a beneficial deviation.

In any case, there is no beneficial deviation for any machine, and $\mathbf{s}^j$ is an equilibrium. □

Theorem 7. *The Price of Anarchy of $\mathcal{SP}_\alpha$ on n machines is at most $(n-1)\alpha + 1$.*

Proof. Fix some underlying $n \times m$ true cost matrix $\mathbf{t}$ and a parameter $\alpha > 1$. Fix also an optimal (makespan-minimizing) allocation OPT of $\mathbf{t}$ and a (pure) Nash equilibrium $\mathbf{s}$ of $\mathcal{SP}_\alpha$

under true costs $\mathbf{t}$. For any task $j = 1, \dots, m$, let $\iota^*(j)$ and $\iota(j)$ denote the machine that gets task j at OPT and $\mathcal{SP}_\alpha(\mathbf{s})$, respectively. Also, let K_i^*, K_i denote the corresponding machine loads and J_i^*, J_i the sets of assigned tasks; that is, for $i = 1, \dots, n$, we define

$$K_i^* \equiv \sum_{j:\iota^*(j)=i} t_{i,j} \qquad \text{and} \qquad K_i \equiv \sum_{j:\iota(j)=i} t_{i,j},$$

where

$$J_i^* = \{j \in J \mid \iota^*(j) = i\} \qquad \text{and} \qquad J_i = \{j \in J \mid \iota(j) = i\}.$$

Finally, it is without loss of generality to assume that $K_1 \geq K_2 \geq \cdots \geq K_n$, so that the makespan of $\mathcal{SP}_\alpha$ on $\mathbf{s}$ is

$$K_1 = \sum_{j \in J_1} t_{1,j} = \sum_{j \in J_1 \cap J_1^*} t_{1,j} + \sum_{j \in J_1 \setminus J_1^*} t_{1,j} \leq \sum_{j \in J_1^*} t_{1,j} + \alpha \sum_{j \in J \setminus J_1^*} t_{\iota^*(j),j},$$

the last inequality holding since, for any task j, $t_{\iota(j),j} \leq \alpha \cdot t_{\iota^*(j),j}$ (due to Lemma 2). Thus, we can bound our mechanism's makespan by

$$K_1 \leq K_1^* + \alpha \sum_{i=2}^{n} K_i^* \leq K_1^* + \alpha(n-1) \max_{i=2,\dots,n} K_i^*.$$

Putting everything together, and denoting for simplicity $x = K_1^*$ and $y = \max_{i=2,\dots,n} K_i^*$, the PoA of $\mathcal{SP}_\alpha$ is finally upper bounded by

$$\mathrm{PoA}(\mathcal{SP}_\alpha) \leq \frac{x + \alpha(n-1)y}{\max_{i=1,\dots,n} K_i^*} = \frac{x + \alpha(n-1)y}{\max\{x, y\}} \leq \alpha(n-1) + 1,$$

the last step coming from applying Lemma 4 with $\beta = \alpha(n-1)$ and $\gamma = 1$. □

Theorem 8. *The Price of Stability of $\mathcal{SP}_\alpha$ on n machines is at most $\frac{n-1}{\alpha} + 1$.*

Proof. Fix some underlying $n \times m$ true cost matrix $\mathbf{t}$ an optimal (makespan-minimizing) allocation OPT of $\mathbf{t}$ and a parameter $\alpha > 1$. Also, let $\iota^*(j)$ denote the machine that gets task j at OPT.

We partition the set of tasks $J = \{1, 2, \dots, m\}$ into two sets J_{small} and J_{large}, based on their processing time under OPT. More specifically, we define:

$$J_{\text{small}} \equiv \{j \in J \mid t_{\iota^*(j),j} \leq \min_{i \in N} t_{i,j}\} \qquad \text{and} \qquad J_{\text{large}} \equiv J \setminus J_{\text{small}} = \{j \in J \mid t_{\iota^*(j),j} > \min_{i \in N} t_{i,j}\}.$$

Intuitively, in light of Lemmas 2 and 3, we can think of J_{small} as containing all the tasks that $\mathcal{SP}_\alpha$ can allocate to the same machine as OPT in some equilibrium and J_{large} containing the tasks for which this is not possible, but which nevertheless end up at a machine that runs them faster than in OPT.

Now consider the pure Nash equilibrium $\mathbf{s}$ of $\mathcal{SP}_\alpha$ (with respect to true running times $\mathbf{t}$) that allocates all small tasks at the same machine as OPT, and all large tasks myopically to the fastest machine for that task. More precisely, if $\iota(j)$ denotes the machine that gets task j under $\mathcal{SP}_\alpha(\mathbf{s})$, we set $\iota(j) = \iota^*(j)$ for all $j \in J_{\text{small}}$ and $\iota(j) \in \operatorname{argmin}_{i \in N} t_{i,j}$ for $j \in J_{\text{large}}$. The fact that such an equilibrium indeed exists, is a consequence of Lemma 3.

Let K_i denote the load of machine i after allocating only the small tasks according to OPT (and thus, also according to the equilibrium $\mathbf{s}$ of $\mathcal{SP}_\alpha$). Let L_i^*, L_i denote the load of machine i

after allocating only the large tasks according to OPT and $\mathbf{s}$, respectively. That is, we formally define:

$$K_i \equiv \sum_{j \in J_{\text{small}}: \iota(j)=i} t_{i,j}, \qquad L_i^* \equiv \sum_{j \in J_{\text{large}}: \iota^*(j)=i} t_{i,j} \qquad \text{and} \qquad L_i \equiv \sum_{j \in J_{\text{large}}: \iota(j)=i} t_{i,j}.$$

Without loss, let's assume that $K_1 \geq K_2 \geq \cdots \geq K_n$. Then, the makespan of OPT is

$$\max_{i \in N}(K_i + L_i^*) \geq \max\left\{K_1 + L_1^*, \max_{i=2,\ldots,n} L_i^*\right\} \geq \max\left\{K_1 + L_1^*, \frac{1}{n-1}\sum_{i=2,\ldots,n} L_i^*\right\},$$

while that of $\mathbf{s}$ can be upper bounded by

$$\max_{i \in N}(K_i + L_i) \leq \max_{i \in N} K_i + \max_{i \in N} L_i \leq K_1 + \sum_{i=1}^n L_i \leq K_1 + \frac{1}{\alpha}\sum_{i=1}^n L_i^* \leq K_1 + L_1^* + \frac{1}{\alpha}\sum_{i=2}^n L_i^*,$$

where the second to last inequality holds due to the fact that for large tasks

$$\sum_{i=1}^n L_i = \sum_{j \in J_{\text{large}}} t_{\iota(j),j} = \sum_{j \in J_{\text{large}}} \min_{i \in N} t_{i,j} \leq \sum_{j \in J_{\text{large}}} \frac{1}{\alpha} t_{\iota^*(j),j} = \frac{1}{\alpha}\sum_{i=1}^n L_i^*,$$

and the last one holds due to $\alpha \geq 1$.

Putting everything together, and denoting for simplicity $x = K_1 + L_1^*$ and $y = \sum_{i=2}^n L_i^*$, we have that

$$\text{PoS}(\mathcal{SP}_\alpha) \leq \frac{x + \frac{1}{\alpha}y}{\max\left\{x, \frac{1}{n-1}y\right\}} \leq \frac{n-1}{\alpha} + 1,$$

the last inequality holding by applying Lemma 4 with $\beta = \frac{1}{\alpha}$ and $\gamma = \frac{1}{n-1}$. □

5 Discussion and Future Directions

In this section, we discuss some implications of our approach, as well as directions for future work. On a general level, one could apply our framework of studying the inefficiency trade-off between the Price of Anarchy and the Price of Stability to many other problems in algorithmic mechanism design, such as auctions [29, 42], machine scheduling without money [23, 20], or resource allocation [11], to name a few, for which the two inefficiency notions have already been studied separately.

In terms of the strategic scheduling setting, our work gives rise to a plethora of intriguing questions for future work, both on a technical and a conceptual level, which we highlight below in more detail.

5.1 General Mechanisms

The major open question associated with our work is whether there exists a mechanism that achieves a better trade-off than that of Theorem 6, or in other words,

> *"Is the yellow region of Fig. 3 empty or not?"*

If such a mechanism exists, it will most probably have to *not* be task-independent[8]; this is somewhat reminiscent of the state-of-the-art results in truthful machine scheduling, where the

[8] See also the discussion in Section 5.2.

best possible mechanisms (with respect to the approximation ratio, see Section 2.2) for several variants of the problem are in fact task-independent [36, 10] and whether a better mechanism that is not task-independent exists is a prominent open question.

While in the case of truthful mechanisms, the general consensus is that the best achievable mechanisms will in fact eventually proven to be task-independent, the situation in the strategic version of the problem might be quite different. This is because we have *all* possible mechanisms at our disposal and it is more conceivable that some allocation rule, tied with some appropriate payment function could potentially outperform the trade-off bounds of Theorem 6. This seems, however, like a quite challenging task; to offer some intuition, we remark the following about the design of general mechanisms for the problem.

The most natural idea is perhaps to use a known algorithm for unrelated machine scheduling [22, 14, 27] such as the greedy allocation algorithm[9] of Ibarra and Kim [22], or even the makespan-optimal algorithm and couple them with a "get-paid-your-load" payment function, where each machine receives a monetary compensation equal to the sum of the reported processing times for the tasks that she gets assigned. This is essentially the generalization of the payment rule of the First-Price mechanism for more general allocation rules. The hope is that by virtue of having a more efficient allocation rule, the resulting mechanism will always have equilibria with small makespan (good PoS) while never having equilibria with very large makespan (good PoA). Unfortunately, one can show that for a large class of such allocation algorithms (which includes all the aforementioned algorithms that have been proposed in the literature for the classical unrelated machine scheduling problem), the Price of Anarchy of the resulting mechanism will be unbounded.

Overall, for a mechanism to lie in the yellow region of Fig. 3, it seems imperative that it will need to employ some more complicated payment function, which will "guide" the agents towards the desired equilibria, rather than simply attempt to implement a better allocation rule with known payment structures.

5.2 The Role of Anonymity

In the proof of Theorem 6, we used the fact that the mechanism in question is anonymous. In many cases in the general related literature, this assumption is without loss of generality, as the best possible mechanisms with respect to an objective are anonymous. In the literature of the unrelated machine scheduling problem in algorithmic mechanism design however, understanding the role of anonymity is a long-standing open problem. In particular, while all the known mechanisms for several variants of the problem are anonymous, the best-known lower bounds for anonymous and non-anonymous mechanisms are strikingly different, from, specifically n in the former case (given by [3] and matching the best-known upper bound of [36]) and 2.61 in the latter case (given by [9]).

In our strategic scheduling problem, anonymity becomes even harder to reason about, since it is a property that applies to the reported strategy profiles and not the underlying true valuations, and the arguments regarding permutations of the underlying true profiles have to be made carefully with regard to the equilibria that they induce, as in our approach in the proof of Theorem 6. Nevertheless, for general (not necessarily anonymous) task-independent mechanisms, we can still show the following inefficiency trade-off. Its proof can be found in Appendix B.

Theorem 9. *For any task-independent scheduling mechanism $\mathcal{M}$ for n machines, and any real*

[9]The algorithm is referred to as "Algorithm D" in [22].

$\alpha > 1$,

$$\text{PoA}(\mathcal{M}) < (n-1)\frac{\alpha}{\sqrt{2}} + 1 \quad \Longrightarrow \quad \text{PoS}(\mathcal{M}) \geq \frac{(n-1)}{\alpha\sqrt{2}} + 1.$$

The above is a strict improvement with respect to the general boundary given by Theorem 2, but still does not quite match the Pareto frontier that we proved for anonymous mechanisms. In other words, the inefficiency boundary given by Theorem 9 lies strictly within the yellow area in Fig. 3. Obtaining tight bounds is an interesting open problem.

5.3 Equilibrium Notion Considerations

In this paper, we study the set of all possible equilibria of mechanisms for the problem, which may include equilibria which are *weakly dominated* (e.g. see [33, Sec. 1.8]), i.e., the agents could use a different strategy instead of their equilibrium strategy and obtain the same utilities, regardless of the reports of the other agents. This type of equilibria are known to exist in the Second-Price mechanism and our class of mechanisms $\mathcal{SP}_\alpha$ also exhibits such equilibria. In order to quantify this type of equilibria in terms of the agents' aversion to risk, Babaioff et al. [4] defined the notion of *exposure factor*, which measures the amount of risk that an agent is willing to expose herself to, when best-responding. In the terminology of our setting, the exposure γ of strategy s_i is such that

$$p_i(s_i, s_{-i}) \geq (1+\gamma)\sum_{i \in S_i} t_i,$$

where s_{-i} is any vector of strategies of the other machines and S_i is the set of tasks assigned to machine i under $\mathbf{x}(s_i, s_{-i})$. In simple words, γ is used to quantify how much extra cost agent i could possibly experience, if all other agents coordinated to a strategy that is the worst possible for the agent. Then, Babaioff et al. [4] proceed to define the set $\mathcal{Q}_t^\gamma$ as the set of all equilibria that consist only of strategies with exposure factor at most γ and the corresponding notion of the Price of Anarchy with the respect to this equilibrium set. Given a parameter α, mechanism $\mathcal{SP}_\alpha$ can be seen as achieving a PoS guarantee even with respect to the set $Q_t^{\alpha-1}$ of equilibria consisting of strategies of exposure at most $\alpha - 1$. Conceptually, even if one is only willing to accept a certain level of risk exposure, an appropriate β can be chosen and the corresponding mechanisms $\mathcal{SP}_\alpha$ for $\alpha \leq \beta$, will lie on the inefficiency boundary, even if the solution concept is the γ-exposure Nash equilibrium.

On the other hand, Theorem 6 holds for any such parameter γ and in particular, it holds for $\gamma = 0$, i.e., when we only consider *undominated* Nash equilibria. Whether there exists a mechanism that can match, in undominated equilibria, the guarantees of $\mathcal{SP}_\alpha$, is an interesting open question; we believe though that this is rather unlikely. That being said, proving impossibility results for general classes of mechanisms seems quite challenging, as the property of not being dominated does not convey much information from a technical standpoint and in particular has different implications for different mechanisms. This is in contrast with the more standard approach in auctions, where *specific* mechanisms have been studied, for which undominated strategies implies a very handy non-overbidding property, e.g., see [16, 30].

5.4 Computational Considerations

As we mentioned earlier, the mechanisms that we construct in this paper (see Definition 5) are rather simple and run in polynomial time, and this is actually the case for all known mechanisms for the truthful scheduling problem as well. It would be interesting to investigate whether adding *computational efficiency* as a desirable property of the mechanisms in question can have any implications on the inefficiency boundary. For truthful scheduling, this is unlikely to be an issue, since the constraint of truthfulness itself typically leads to rather simple mechanisms which are easily seen to be efficient. As we hinted in Section 5.1 however, it is conceivable that from the space of all possible mechanisms that we can use, the best one might employ an allocation algorithm that is computationally intractable. Concretely, it could be possible that the makespan-optimal algorithm (which does not run in polynomial time, unless P=NP [27]) can be coupled with an appropriate payment function to achieve a better trade-off guarantee. From this discussion, we deduce the following, very interesting question:

> *"If we add computational efficiency as a constraint, can we prove a stronger inefficiency boundary than that of Fig. 2?."*

Conceptually, the question above regards whether there is a fundamental connection between the running time of the allocation rule and the PoA/PoS trade-off that can be explored via a corresponding inefficiency boundary.

Acknowledgements We are grateful to Elias Koutsoupias and Maria Kyropoulou for useful discussions.

References

[1] E. Anshelevich, A. Dasgupta, J. Kleinberg, É. Tardos, T. Wexler, and T. Roughgarden. The price of stability for network design with fair cost allocation. *SIAM Journal on Computing*, 38(4):1602–1623, 2008.

[2] A. Archer and É. Tardos. Truthful mechanisms for one-parameter agents. In *Proceedings of the 42nd IEEE symposium on Foundations of Computer Science*, pages 482–491. IEEE, 2001.

[3] I. Ashlagi, S. Dobzinski, and R. Lavi. Optimal lower bounds for anonymous scheduling mechanisms. *Mathematics of Operations Research*, 37(2):244–258, 2012.

[4] M. Babaioff, B. Lucier, N. Nisan, and R. Paes Leme. On the efficiency of the Walrasian mechanism. In *Proceedings of the 15th ACM Conference on Economics and Computation*, pages 783–800, 2014.

[5] I. Caragiannis, C. Kaklamanis, P. Kanellopoulos, M. Kyropoulou, B. Lucier, R. P. Leme, and É. Tardos. Bounding the inefficiency of outcomes in generalized second price auctions. *Journal of Economic Theory*, 156:343–388, 2015.

[6] S. Chawla, J. D. Hartline, D. Malec, and B. Sivan. Prior-independent mechanisms for scheduling. In *Proceedings of the 45th Annual ACM Symposium on Theory of Computing*, pages 51–60, 2013.

[7] G. Christodoulou and E. Koutsoupias. Mechanism design for scheduling. *Bulletin of the EATCS*, 97:40–59, 2009.

[8] G. Christodoulou, E. Koutsoupias, and A. Vidali. A characterization of 2-player mechanisms for scheduling. In *16th Annual European Symposium on Algorithms*, pages 297–307, 2008.

[9] G. Christodoulou, E. Koutsoupias, and A. Vidali. A lower bound for scheduling mechanisms. *Algorithmica*, 55(4):729–740, 2009.

[10] G. Christodoulou, E. Koutsoupias, and A. Kovács. Mechanism design for fractional scheduling on unrelated machines. *ACM Trans. Algorithms*, 6(2):38:1–38:18, 2010.

[11] G. Christodoulou, A. Filos-Ratsikas, S. K. S. Frederiksen, P. W. Goldberg, J. Zhang, and J. Zhang. Social welfare in one-sided matching mechanisms (extended abstract). In *Proceedings of the 2016 International Conference on Autonomous Agents and Multiagent Systems*, pages 1297–1298, 2016.

[12] E. H. Clarke. Multipart pricing of public goods. *Public Choice*, 11(1):17–33, 1971.

[13] C. Daskalakis and S. M. Weinberg. Bayesian truthful mechanisms for job scheduling from bi-criterion approximation algorithms. In *Proceedings of the 26th annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1934–1952, 2015.

[14] E. Davis and J. M. Jaffe. Algorithms for scheduling tasks on unrelated processors. *J. ACM*, 28(4):721–736, 1981.

[15] S. Dobzinski and M. Sundararajan. On characterizations of truthful mechanisms for combinatorial auctions and scheduling. In *Proceedings of the 9th ACM Conference on Electronic Commerce*, pages 38–47, 2008.

[16] M. Feldman, B. Lucier, and N. Nisan. Correlated and coarse equilibria of single-item auctions. In *International Conference on Web and Internet Economics*, pages 131–144. Springer, 2016.

[17] A. Filos-Ratsikas, S. K. S. Frederiksen, and J. Zhang. Social welfare in one-sided matchings: Random priority and beyond. In R. Lavi, editor, *Algorithmic Game Theory*, pages 1–12. Springer, 2014.

[18] M. Gairing. Covering games: Approximation through non-cooperation. In *International Workshop on Internet and Network Economics*, pages 184–195. Springer, 2009.

[19] Y. Giannakopoulos and M. Kyropoulou. The VCG mechanism for bayesian scheduling. *ACM Trans. Econ. Comput.*, 5(4):19:1–19:16, 2017.

[20] Y. Giannakopoulos, E. Koutsoupias, and M. Kyropoulou. The anarchy of scheduling without money. In *Algorithmic Game Theory: 9th International Symposium*, pages 302–314, 2016.

[21] T. Groves. Incentives in teams. *Econometrica*, 41(4):617–631, 1973.

[22] O. H. Ibarra and C. E. Kim. Heuristic algorithms for scheduling independent tasks on nonidentical processors. *J. ACM*, 24(2):280–289, 1977.

[23] E. Koutsoupias. Scheduling without payments. *Theory of Computing Systems*, 54(3): 375–387, 2014.

[24] E. Koutsoupias and C. Papadimitriou. Worst-case equilibria. *Computer Science Review*, 3 (2):65–69, 2009.

[25] E. Koutsoupias and A. Vidali. A lower bound of $1+\varphi$ for truthful scheduling mechanisms. *Algorithmica*, 66(1):211–223, 2013.

[26] R. Lavi and C. Swamy. Truthful mechanism design for multidimensional scheduling via cycle monotonicity. *Games and Economic Behavior*, 67(1):99–124, 2009.

[27] J. K. Lenstra, D. B. Shmoys, and É. Tardos. Approximation algorithms for scheduling unrelated parallel machines. *Mathematical Programming*, 46(1):259–271, 1990.

[28] P. Lu and C. Yu. Randomized truthful mechanisms for scheduling unrelated machines. In *International Workshop on Internet and Network Economics*, pages 402–413. Springer, 2008.

[29] B. Lucier, Y. Singer, V. Syrgkanis, and É. Tardos. Equilibrium in combinatorial public projects. In Y. Chen and N. Immorlica, editors, *Web and Internet Economics*, pages 347–360. Springer, 2013.

[30] E. Markakis and O. Telelis. Uniform price auctions: Equilibria and efficiency. *Theory of Computing Systems*, 57(3):549–575, 2015.

[31] A. Mu'alem and M. Schapira. Setting lower bounds on truthfulness. In *Proceedings of the 18th Annual ACM-SIAM Symposium on Discrete algorithms*, pages 1143–1152, 2007.

[32] R. B. Myerson. Optimal auction design. *Mathematics of Operations Research*, 6(1):58–73, 1981.

[33] R. B. Myerson. *Game Theory: Analysis of Conflict*. Harvard University Press, 1997.

[34] N. Nisan. Introduction to Mechanism Design (for Computer Scientists). In N. Nisan, T. Roughgarden, É. Tardos, and V. Vazirani, editors, *Algorithmic Game Theory*, chapter 9. Cambridge University Press, 2007.

[35] N. Nisan and A. Ronen. Algorithmic mechanism design. *Games and Economic Behavior*, 35(1/2):166–196, 2001.

[36] N. Nisan and A. Ronen. Computationally feasible VCG mechanisms. *J. Artif. Int. Res.*, 29(1):19–47, 2007.

[37] V. Ramaswamy, D. Paccagnan, and J. R. Marden. Multiagent coverage problems: The trade-off between anarchy and stability. *CoRR*, abs/1710.01409, July 2018. URL http://arxiv.org/abs/1710.01409.

[38] T. Roughgarden and É. Tardos. How bad is selfish routing? *J. ACM*, 49(2):236–259, 2002.

[39] T. Roughgarden, V. Syrgkanis, and E. Tardos. The price of anarchy in auctions. *Journal of Artificial Intelligence Research*, 59:59–101, 2017.

[40] M. Saks and L. Yu. Weak monotonicity suffices for truthfulness on convex domains. In *Proceedings of the 6th ACM Conference on Electronic Commerce*, pages 286–293, 2005.

[41] A. S. Schulz and N. S. Moses. On the performance of user equilibria in traffic networks. In *Proceedings of the Fourteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 86–87, 2003.

[42] V. Syrgkanis and E. Tardos. Composable and efficient mechanisms. In *Proceedings of the 45th Annual ACM Symposium on Theory of Computing*, pages 211–220, 2013.

[43] W. Vickrey. Counterspeculation, auctions and competitive sealed tenders. *Journal of Finance*, 16(1):8–37, 1961.

Appendix

A Technical Lemmas

Lemma 4. *For any nonnegative reals x, y with $xy \neq 0$ and all positive reals β, γ:*

$$\frac{x + \beta y}{\max\{x, \gamma y\}} \leq \frac{\beta}{\gamma} + 1$$

Proof. There are two cases to consider. First, if $x \geq \gamma y$, then

$$\frac{x + \beta y}{\max\{x, \gamma y\}} = \frac{x + \beta y}{x} = \beta \frac{y}{x} + 1 \leq \beta \frac{1}{\gamma} + 1.$$

Secondly, if $\gamma y \geq x$, then

$$\frac{x + \beta y}{\max\{x, \gamma y\}} = \frac{x + \beta y}{\gamma y} = \frac{\beta}{\gamma} + \frac{x}{\gamma y} \leq \frac{\beta}{\gamma} + 1.$$

□

Lemma 5. *For $i, j = 1, \ldots, n$, let reals $\alpha > 0$, $a_{i,j} > 0$ (for $i \neq j$) and $a_{i,i} = 0$ such that for all j:*

$$\sum_{i=1}^{n} a_{i,j} < \frac{(n-1) \cdot \alpha}{\sqrt{2}}.$$

Then, for any positive $\varepsilon \leq \frac{\alpha}{(n-1)\sqrt{2}}$ there exists some i such that:

$$\max_{\emptyset \neq J \subseteq [n] \setminus \{i\}} \frac{|J|}{\max_{j \in J} a_{i,j} + \varepsilon} > \frac{n-1}{\alpha\sqrt{2}}.$$

The above inequality cannot be further improved.

Proof. We proceed using a proof by contradiction. For any fixed i, we use the index i_k to refer to a_{i,i_k}, the k-th smallest among the $a_{i,j}$. By contradiction, we have that for all i and nonempty $J \subseteq [n] \setminus \{i\}$:

$$\frac{|J|}{\max_{j \in J} a_{i,j} + \varepsilon} \leq \frac{n-1}{\alpha\sqrt{2}}.$$

In particular, for all $J_k = \{2, \ldots, k\}$ (note that $a_{i,i_1} = a_{i,i} = 0$) we get:

$$\frac{k-1}{a_{i,i_k} + \varepsilon} \leq \frac{n-1}{\alpha\sqrt{2}} \quad \Longrightarrow \quad a_{i,i_k} \geq \alpha \frac{(k-1)\sqrt{2}}{n-1} - \varepsilon.$$

Summing over all values of i and j:

$$\begin{aligned}
\sum_{i=1}^{n} \sum_{j=1}^{n} a_{i,j} = \sum_{i=1}^{n} \sum_{k=2}^{n} a_{i,i_k} &\geq \sum_{i=1}^{n} \sum_{k=2}^{n} \alpha \frac{(k-1)\sqrt{2}}{n-1} - \varepsilon \\
&= \frac{\alpha\sqrt{2}}{n-1} \sum_{i=1}^{n} \sum_{k=2}^{n} k - 1 - \varepsilon \\
&= \frac{\alpha\sqrt{2}}{n-1} \cdot n \cdot \frac{n(n-1)}{2} - n(n-1)\varepsilon \\
&= n \frac{n \cdot \alpha}{\sqrt{2}} - n(n-1)\varepsilon.
\end{aligned}$$

Since the a_{ij} are partitioned by the n distinct values of j, there must be some j for which:

$$\sum_{i=1}^{n} a_{i,j} \geq \frac{n \cdot \alpha}{\sqrt{2}} - (n-1)\varepsilon \geq \frac{(n-1) \cdot \alpha}{\sqrt{2}},$$

leading to a contradiction.

This result is essentially tight. For any $\delta > 0$ consider the matrix:

$$\alpha \frac{\sqrt{2} - \delta}{n-1} \cdot \begin{pmatrix} 0 & 1 & 2 & \cdots & n-2 & n-1 \\ n-1 & 0 & 1 & \cdots & n-3 & n-2 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & 2 & 3 & \cdots & n-1 & 0 \end{pmatrix},$$

where every column sum is $< (n-1)\alpha/\sqrt{2}$ and for every row i and nonempty $J \subseteq [n] \setminus \{i\}$:

$$\frac{|J|}{\max_{j \in J} a_{i,j}} \leq \frac{|J|}{a_{i,i_{|J|+1}}} = \frac{|J|}{\alpha(\sqrt{2} - \delta)\frac{|J|}{n-1}} = \frac{n-1}{\alpha(\sqrt{2} - \delta)}.$$

□

B Proof of Theorem 9

Without loss of generality, we assume that mechanism $\mathcal{M}$ allocates each task-independently by running the *same* single-task mechanism for *every* task. The reason the analysis carries over, is that we will only use tasks drawn from a finite pool. Restricted to these profiles of true processing times, there are only finitely many *essentially different* single-task mechanisms, when the difference is measured from the perspective of allocations. Therefore, even if the task-independent mechanism $\mathcal{M}$ used a different one for every task, we could always find n single-component mechanisms operating the same way. For a rigorous treatment, please we refer the reader to Lemma 6.

We aim to identify some "weakness" of $\mathcal{M}$ by discovering, for canonical instances that are permutations of the true processing times $(1, x, \infty, \infty, \ldots)$, just how much larger x can get, so that there exists some equilibrium under which the task is allocated to the slow machine. We refer to the machine with processing time 1 as the *fast* machine and to the other, which does not have a processing time of ∞, as the *slow* machine. As the mechanism is *not* anonymous, there might be a different x for every permutation.

Formally, fix $\frac{\alpha}{(n-1)\sqrt{2}} \geq \varepsilon > 0$ and for $i, j = 1, \ldots, n$ and $i \neq j$, let

$$a_{i,j} = \max_{k \in \mathbb{N}} \left\{ k \cdot \varepsilon \;\middle|\; \text{for } \mathbf{t}_{i,j} = (\ldots, \underbrace{1}_{i}, \ldots, \underbrace{k \cdot \varepsilon}_{j}, \ldots),\ \exists \mathbf{s}_{i,j} \text{ s.t. machine } j \text{ is allocated the task} \right\}$$

be the maximum processing time of the slow machine, when the fast machine has index i, the slow machine has index j and the slow machine can still receive the task for some equilibrium. To ensure that Lemma 6 can be applied, we need to discretize the processing times, in order to have a finite number of possible tasks. Let $\bar{a}_{i,j} = a_{i,j} + \varepsilon$. Clearly, if $a_{i,j}$ is replaced by $\bar{a}_{i,j}$ in its canonical instance then the *only allocation* remaining will be to give the task to the fast machine. We call this instance $\bar{\mathbf{t}}_{i,j}$. For convenience, we also set $a_{i,i} = \bar{a}_{i,i} = 0$.

Next, observe that $0 < a_{i,j} < (n-1)\alpha/\sqrt{2}$ for all $i \neq j$. The upper bound is due to the assumption on the PoA, which would otherwise be violated for $\mathbf{t}_{i,j}$. For the lower bound, the instance $\bar{\mathbf{t}}_{i,j}$ would have PoA $\geq 1/\varepsilon$, leading to a contradiction, for some small enough ε.

Fixing j, we create the following profile of true processing times:

$$\mathbf{t}^j = (\mathbf{t}_{1,j}, \dots, \mathbf{t}_{(i-1),j}, \mathbf{f}^\star, \mathbf{t}_{(i+1),j}, \dots, \mathbf{t}_{n,j}),$$

where $\mathbf{f}^\star$ contains only one entry that is not ∞, at row j. Presented in a more convenient matrix form:

$$\left.\begin{pmatrix} 1 & \infty & \infty & \infty & \infty & \cdots & \infty & \infty & \infty \\ \infty & 1 & \infty & \infty & \infty & \cdots & \infty & \infty & \infty \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \infty \\ \infty & \infty & \cdots & 1 & \infty & \infty & \cdots & \infty & \infty \\ a_{1,j} & a_{2,j} & \cdots & a_{(j-1),j} & 1 & a_{(j+1),j} & \cdots & a_{(n-1),j} & a_{n,j} \\ \infty & \infty & \cdots & \infty & \infty & 1 & \infty & \cdots & \infty \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \ddots & \ddots & \vdots \\ \infty & \infty & \cdots & \infty & \infty & \infty & \infty & 1 & \infty \\ \infty & \infty & \cdots & \infty & \infty & \infty & \infty & \infty & 1 \end{pmatrix}\right\} n,$$

where machine j is the slow machine for all the tasks and for each task there exists exactly one, distinct, fast machine. By task-independence, the profile of reported processing times $(\mathbf{s}_{1,j}, \mathbf{s}_{2,j}, \dots, \mathbf{s}_{(i-1),j}, \mathbf{f}^\star_{EQ}, \mathbf{s}_{(i+1),j}, \dots, \mathbf{s}_{n,j})$ is an equilibrium in which all tasks are allocated to the slow machine and $\mathbf{f}^\star_{EQ}$ is any equilibrium for task j, which clearly has to select the j-th machine. Since the optimal makespan is 1, by the given bound on the PoA, we have that for all j:

$$\sum_{i=1}^{n} a_{i,j} + 1 < \frac{(n-1)\cdot\alpha}{\sqrt{2}} + 1 \Rightarrow \sum_{i=1}^{n} a_{i,j} < \frac{(n-1)\cdot\alpha}{\sqrt{2}} \tag{2}$$

In the proof of Theorem 6, in order to obtain the bound fo the PoS, we swapped the positions of the slow and fast machines on each column, which was enabled by the assumption that the mechanism there was anonymous. Since mechanism $\mathcal{M}$ is not anonymous, here we have to make a slight adjustment and also change $a_{i,j}$ to $\bar{a}_{i,j}$. To obtain a lower bound on the PoS, we instead fix i and using the true processing times

$$\bar{\mathbf{t}}^j = (\bar{\mathbf{t}}_{i,1}, \dots, \bar{\mathbf{t}}_{i,(i-1)}, \bar{\mathbf{f}}^\star, \bar{\mathbf{t}}_{i,(i+1)}, \dots, \bar{\mathbf{t}}_{i,n}),$$

we get a matrix similar to the previous one, i.e.,

$$\begin{pmatrix} \bar{a}_{i,1} & \infty & \infty & \infty & \infty & \cdots & \infty & \infty & \infty \\ \infty & \bar{a}_{i,2} & \infty & \infty & \infty & \cdots & \infty & \infty & \infty \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \infty \\ \infty & \infty & \cdots & \bar{a}_{i,(i-1)} & \infty & \infty & \cdots & \infty & \infty \\ 1 & 1 & \cdots & 1 & \max_j \bar{a}_{i,j} & 1 & \cdots & 1 & 1 \\ \infty & \infty & \cdots & \infty & \infty & \bar{a}_{i,(i+1)} & \infty & \cdots & \infty \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \ddots & \ddots & \vdots \\ \infty & \infty & \cdots & \infty & \infty & \infty & \infty & \bar{a}_{i,(n-1)} & \infty \\ \infty & \infty & \cdots & \infty & \infty & \infty & \infty & \infty & \bar{a}_{i,n} \end{pmatrix}$$

Again, by task independence and the definition of $\bar{a}_{i,j}$, the mechanism has to allocate each task to the *fast* machine, which for all tasks is machine i. The optimal allocation could potentially only allocate to slow machines and produce an allocation with makespan at most $\max_j \bar{a}_{i,j}$.

The lower bound on the PoS can be further improved by restricting the input to only contain a nonempty subset $J \subseteq [n] \setminus \{i\}$ of the tasks, plus task i which is special. This reduces the makespan of *both* $\mathcal{M}$ and the optimal, but the overall fraction could increase. Notice that task i (which has only one entry which is not ∞, adjusted to be the maximum amongst the $|J|$ selected $\bar{a}_{i,j}$) is *always* added, as it increases $\mathcal{M}$'s makespan for free. In particular:

$$\text{PoS} \geq \max_{J\subseteq[n]\setminus\{i\}} \frac{|J| + \max_{j\in J} \bar{a}_{i,j}}{\max_{j\in J} \bar{a}_{i,j}} = \max_{J\subseteq[n]\setminus\{i\}} \frac{|J|}{\max_{j\in J} \bar{a}_{i,j}} + 1 \tag{3}$$

Notice that the $a_{i,j}$ satisfy the premises of Lemma 5. Applying it to Eq. (3) we get:

$$\text{PoS} \geq \max_{J\subseteq[n]\setminus\{i\}} \frac{|J|}{\max_{j\in J} \bar{a}_{i,j}} + 1 = \max_{J\subseteq[n]\setminus\{i\}} \frac{|J|}{\max_{j\in J} a_{i,j} + \varepsilon} + 1 > \frac{n-1}{\alpha\sqrt{2}} + 1.$$

This completes the proof.

The proof of Theorem 9 shows that foregoing anonymity cannot lead to mechanisms with asymptotically tighter Pareto frontiers. Moreover, it leads the the following observations. Most likely, if there exists a stronger bound, it cannot be obtained with our canonical tasks and would need to use a richer input. At the same time, if a non-anonymous mechanism that achieves a better trade-off than that given by Theorem 6 does exist, the matrix at the end of Lemma 5 could provide some insight on what that mechanism could look like, at least when restricted to these tasks.

Lemma 6. *Theorem 9 holds for all task-independent mechanisms $\mathcal{M}$.*

Proof. Continuing from our note in the beginning of Theorem 9, we show that given a task-independent mechanism $\mathcal{M}$, it is always possible to find n single-task component mechanisms that behave the same way for the very specific inputs needed for the lower bounds. Notice that in the proof of Theorem 9 the tasks we used all come from a finite set of true processing times $\mathcal{C}$. In particular, $\mathcal{C}$ contains all the permutations of $(1, a, \infty, \infty, \ldots, \infty)$, where a is of the form $k \cdot \varepsilon$ and $0 < a < (n-1) \cdot \alpha/\sqrt{2} + \varepsilon$. Therefore:

$$|\mathcal{C}| = n(n-1)\frac{(n-1) \cdot \alpha/\sqrt{2}}{\varepsilon}.$$

Fix a single-task mechanism $\mathcal{A}$. For a true processing time vector $\mathbf{t} \in \mathcal{C}$, call machine j *accessible* from $\mathbf{t}$ if there exists an equilibrium for which the task is assigned to that machine. Let $ACC(\mathbf{t})$ be the set of accessible machines for component $\mathcal{A}$ for that $\mathbf{t}$ and $\mathcal{B}(\mathcal{A})$ be the *behaviour* set of $\mathcal{A}$, defined as

$$\mathcal{B}(\mathcal{A}) = \{(\mathbf{t}, ACC(\mathbf{t})) \mid \mathbf{t} \in \mathcal{C}\}.$$

This fully characterizes $\mathcal{A}$ from the perspective of accessible allocations for tasks in $\mathcal{C}$. Note that for two different single-task mechanisms $\mathcal{A}_1$ and $\mathcal{A}_2$, it could be that $\mathcal{B}(\mathcal{A}_1) = \mathcal{B}(\mathcal{A}_2)$ without them being the same mechanism: they must however reach the same allocations given the same $\mathbf{t}$ (potentially through different equilibria).

Clearly, $ACC(\mathbf{t})$ can be one of $2^n - 1$ possible sets. This is because $ACC(\mathbf{t})$ can be any subset of the n machines, except the empty subset, since each task has to be allocated to some machine at every equilibrium. This is however all we need in Theorem 9. Therefore, the total number of behaviour sets is at most:

$$(2^n - 1)^{|\mathcal{C}|},$$

which is quite large, but bounded.

Let $\mathcal{A}_j$ be the single-task mechanisms used by mechanism $\mathcal{M}$ for the j'th task, given an input with $n(2^n - 1)^{|\mathcal{C}|}$ tasks from $\mathcal{C}$ in total. By the pidgeonhole principle, there must be at least n such single-task mechanisms with the same behaviour set. Setting all other tasks to have real processing times 0 for all tasks, we have successfully extracted a mechanism $\mathcal{M}'$ that behaves exactly as needed for Theorem 9. □