OR Spectrum (2018) 40:583–611
https://doi.org/10.1007/s00291-018-0512-8

REGULAR ARTICLE

Mechanism design for machine scheduling problems: classification and literature overview

Dominik Kress[1] · **Sebastian Meiswinkel**[1] · **Erwin Pesch**[1,2]

Received: 13 December 2016 / Accepted: 2 February 2018 / Published online: 27 February 2018

Abstract This paper provides a literature overview on (direct revelation) algorithmic mechanism design in the context of machine scheduling problems. Here, one takes a game-theoretic perspective and assumes that part of the relevant data of the machine scheduling problem is private information of selfish players (usually machines or jobs) who may try to influence the solution determined by the scheduling algorithm by submitting false data. A central planner is in charge of controlling and designing the algorithm and a rewarding scheme that defines payments among planner and players based on the submitted data. The planner may, for example, want to design algorithm and payments such that reporting the true data always maximizes the utility functions of rationally acting players, because this enables the planner to generate fair solutions with respect to some social criterion that considers the interests of all players. We review the categories and characterizing problem features of machine scheduling settings in the algorithmic mechanism design literature and extend the widely accepted classification scheme of Graham et al. (Ann Discrete Math 5:287–326, 1979) for scheduling problems to include aspects relating to mechanism design. Based on this hierarchical scheme, we give a systematic overview of recent contributions in this field of research.

✉ Dominik Kress
dominik.kress@uni-siegen.de

Sebastian Meiswinkel
sebastian.meiswinkel@uni-siegen.de

Erwin Pesch
erwin.pesch@uni-siegen.de

1 Management Information Science, University of Siegen, Kohlbettstr. 15, 57068 Siegen, Germany

2 Center for Advanced Studies in Management, HHL Leipzig, Jahnallee 59, 04109 Leipzig, Germany

Keywords Algorithmic mechanism design · Machine scheduling · Game theory · Incentive compatibility

1 Introduction and scope of review

There exists a tremendous body of literature that focuses on intersections of (algorithmic aspects of) computer science and game theory (as well as economic theory). The resulting fields of intersecting disciplines are usually referred to as *algorithmic game theory* (an excellent introduction and overview is given by Nisan et al. 2007). Many research articles in this field focus on auction contexts (see Krishna 2010). Recently, however, there has been a growing interest in taking a game-theoretic perspective on machine scheduling problems, which has resulted in a fairly large amount of research articles that we aim to review and classify in this article.

Broadly speaking, *scheduling problems* are concerned with allocating scarce resources over time to perform a set of tasks with the objective of optimizing one or more performance measures (Błażewicz et al. 2007; Leung 2004). Resources, tasks and performance measures can be of very different nature. As indicated above, we will focus on resources that (directly) represent some kind of processor or machine, i.e., machine scheduling problems, and set the scope of our literature review, which complements the articles by Heydenreich et al. (2007) and Christodoulou and Koutsoupias (2009), to include research on

- *machine scheduling* problems
- in *offline-settings*, where all information regarding the problem is known or has been announced (see below) at the unique time of planning,
- in a *noncooperative* game-theoretic context, where players cannot form coalitions
- and have *private information* on their own characteristics which they directly (but not necessarily truthfully) announce by making a single claim,
- in the presence of a *central authority* that is in charge of designing a rewarding scheme and the scheduling algorithm that determines the final schedule based on the information submitted by the players and the publicly known information.

In order to give a systematic record of the academic efforts in the above field of research, we provide a corresponding hierarchical classification scheme. This scheme augments the classification scheme by Graham et al. (1979) for machine scheduling problems, which is widely used and generally accepted in the scheduling community. We are motivated by the fact that adoptions and extensions of Graham et al. (1979) have been successfully implemented in a variety of other problem fields (see, for example, Allahverdi et al. 2008; Boysen and Fliedner 2010; Boysen et al. 2007, 2009; Brucker et al. 1999; Potts and Kovalyov 2000).

1.1 Scope of review

In this section, we present details on the scope of our literature review and additionally introduce the notation used throughout this article. For the sake of brevity, we assume the reader to be familiar with the basic theory of machine scheduling problems and

Table 1 Notation: machine scheduling problems

J	Set of jobs	$\|J\| = n$
M	Set of machines	$\|M\| = m$
O	Set of feasible solutions of scheduling problem	
t_{ij}	Processing time of job $j \in J$ on machine $i \in M$	
	• Single machine or identical (parallel) machines: $t_{ij} = t_j \; \forall i \in M$	
	• Uniform (parallel) machines: $t_{ij} = t_j/s_i$ for a given speed s_i	
	• Unrelated (parallel) machines: $t_{ij} = t_j/s_{ij}$ for a given speed s_{ij}	
w_j	Weight of job $j \in J$	
r_j	Release date of job $j \in J$	
d_j	Due date or deadline of job $j \in J$	
C_j	Completion time of job $j \in J$	$C_j : O \rightarrow \mathbb{R}_{\geq 0}$
U_j	Unit penalty of job $j \in J$: 1 if j completes strictly after d_j, 0 otherwise	$U_j : O \rightarrow \{0, 1\}$
L_i	Load of machine $i \in M$	$L_i : O \rightarrow \mathbb{R}_{\geq 0}$

the main concepts of *game theory* and refer to Błażewicz et al. (2007), Leung (2004), and Fudenberg and Tirole (1991) for comprehensive introductions to these fields of research.

1.1.1 Machine scheduling problems

The notation used in the scheduling and (algorithmic) game theory communities is not always compatible. Therefore, we will sometimes deviate from the standard scheduling notation. Basically, a machine scheduling problem is characterized by a set J of n jobs (tasks) and a set M of m machines. A feasible schedule o assigns machines of the set M to jobs of the set J in order to complete all jobs under a set of imposed constraints. We denote the set of all feasible schedules of a given machine scheduling problem by O. Jobs and machines are characterized by certain parameters, e.g., processing times or speeds. Furthermore, there exist different performance measures. We assume the reader to be aware of the corresponding definitions and restrict ourselves to giving an overview of the notation relevant for this article in Table 1.

As mentioned above, Graham et al. (1979) present a widely used and generally accepted *classification scheme* for machine scheduling problems. It represents a specific problem by a three-field notation, $\alpha|\beta|\gamma$, where α describes the machine environment, β refers to job characteristics, and γ relates to the (global) performance measure (optimality criterion). Each field of the triple includes multiple elements, e.g., $\alpha = \alpha_1, \alpha_2, \ldots$, which represent specific problem properties. The empty symbol, $\circ$, denotes the default value of an element and is skipped when a triple is actually specified.

1.1.2 Algorithmic game theory

The games considered throughout this paper have three basic elements: players, strategy spaces, and utility functions. Furthermore, we will restrict ourselves to considering

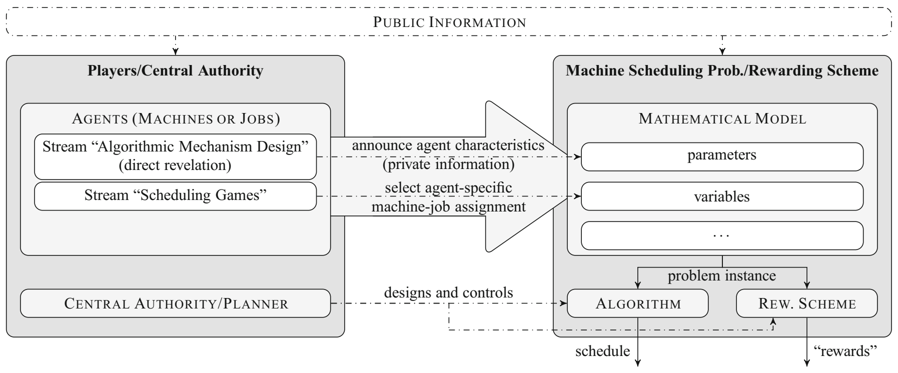
Public Information
Players/Central Authority
Agents (Machines or Jobs)
Stream “Algorithmic Mechanism Design” (direct revelation)
Stream “Scheduling Games”
Central Authority/Planner
announce agent characteristics (private information)
select agent-specific machine-job assignment
designs and controls
Machine Scheduling Prob./Rewarding Scheme
Mathematical Model
parameters
variables
...
problem instance
Algorithm
Rew. Scheme
schedule
“rewards”

Fig. 1 Algorithmic mechanism design (in case of direct revelation) and scheduling games

noncooperative games. That is, players cannot form coalitions in order to generate group decisions. In the context of machine scheduling problems, *players* may be machines or jobs. More generally, one may also think of “owners” of multiple machines or jobs that act as single players. Each player has an associated *strategy space* that represents the options that the player can select from when the game is played. For example, when the players correspond to jobs, each job may be allowed to select a machine to be processed on. A player’s *utility function* assigns a utility level to every vector of strategies, i.e., each combination of strategies that can potentially be selected by all players. With respect to machine scheduling problems, the utility level could, for instance, correspond to the completion time of a given job.

We will consider fairly specific problem settings in the field of algorithmic game theory for machine scheduling problems. These settings are characterized by the existence of (rational and selfish) players, who are typically referred to as *agents* and can make a single claim on some pieces of information that may affect the final schedule. Furthermore, there exists a *central authority/planner*, who is in charge of designing an interaction protocol, a rewarding scheme (e.g., payments among players), and a scheduling algorithm that determines the final schedule. Within this scope, there are two main streams of literature that differ in the type of information that the agents possess and in the way that the information affects an instance of the considered machine scheduling problem (Fig. 1).

1. In this article, we will focus on one of these streams, which presumes that the agents have *private information* on some of their own characteristics. Jobs, for instance, may privately know their due dates or job weights. The remaining data, e.g., the number of machines and jobs, is usually assumed to be publicly known. The central planner designs some protocol of interaction that the agents have to follow. This protocol may be fairly general. We will, however, restrict ourselves to considering “direct protocols” that allow the agents to solely (but not necessarily truthfully) announce concrete values that represent their private information when the game begins. In terms of optimization problems, these agents therefore fix a subset of parameters. When acting selfishly, they may try to influence the solution determined by the scheduling algorithm by submitting false information if this can

increase their utility. However, by designing appropriate algorithms and rewarding schemes that set the right incentives, the central planner can extract the true information of these players, for example, in order to generate fair solutions with respect to some social criterion that considers the interests of all agents.

2. In the second stream, which we exclude from our review for the sake of brevity (the interested reader may refer to Heydenreich et al. 2007), the (usually completely informed) agents, again pursuing selfish goals, commit *decisions* on machine-job assignments and thus implicitly fix variables of optimization problems. We can, for example, think of jobs whose strategy spaces correspond to the set of machines, i.e., jobs who can choose to be processed on specific machines.

We would like to stress the fact that the aforementioned fields of research are not always clearly separated in the literature. Similarly, the terms used to identify specific problems within these fields may differ among different articles. We will follow Nisan and Ronen (2001), who define *(algorithmic) mechanism design* to aim at "study[ing] how privately known preferences […] can be aggregated toward a 'social choice' " (see also Nisan and Ronen 1999), which corresponds to the first stream described above. Our focus on direct protocols is usually termed *direct revelation* (see, for example, Nisan 2007). Others use the term "algorithmic mechanism design" in a more general context, even when there is no privately owned information (see, for instance, Immorlica et al. 2009). Problems in the second stream are sometimes referred to as *(machine) scheduling games* (see, for instance, Harks et al. 2011; Roughgarden and Tardos 2007) or *load balancing games* (Vöcking 2007). These games are closely related to the categories of *congestion games* (Rosenthal 1973) and *coordination mechanisms* (Christodoulou et al. 2009a). In all of these areas, one is usually interested in deciding whether (Nash) equilibria exist, how (in-)efficient these equilibria are when compared to socially optimal solutions, and how fast algorithms can compute them (Harks et al. 2011; Roughgarden and Tardos 2007).

1.1.3 Algorithmic mechanism design

Based on the illustration in Fig. 2, we will now describe the (direct revelation) algorithmic mechanism design setting in the context of machine scheduling problems in more detail. The corresponding notation used throughout this article is summarized in Table 2.

Let A denote the set of rational and selfish agents. Each agent $k \in A$ has a (true) *valuation function* $v_k^t : O \rightarrow \mathbb{R}$ that maps every feasible schedule of the considered scheduling problem to a real value. v_k^t is private information of the agent and is thus sometimes referred to as the agent's *type*. Negative values can, for example, relate to costs incurred to a (job) agent due to waiting for being completed.

Each agent $k \in A$ reports a *valuation function* v_k that may deviate from the true valuation function v_k^t to the mechanism. Each valuation function $v_k, k \in A$, is element of a publicly known set V_k. We define $V := V_1 \times \cdots \times V_{|A|}$. Furthermore, we denote the vector of all valuation functions reported to the mechanism by $v = (v_1, \ldots, v_{|A|})$ and the vector of all valuation functions reported to the mechanism except of v_k by

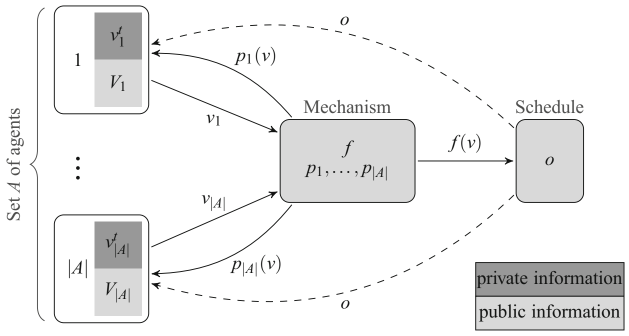

Fig. 2 (Direct revelation) algorithmic mechanism design. (Reproduced with permission from Kress et al. 2017)

Table 2 Notation: algorithmic mechanism design

A	Set of agents	
V_k	Set of potential valuation functions for agent $k \in A$	
V	Cartesian product of sets $V_k, k \in A$	$V = V_1 \times \cdots \times V_{\|A\|}$
V_{-k}	Cartesian product of sets $V_l, l \in A \setminus \{k\}$	$V = V_1 \times \cdots \times V_{k-1} \times V_{k+1} \times \cdots \times V_{\|A\|}$
f	Social choice function/allocation rule	$f : V \rightarrow O$
v_k^t	True valuation function of agent $k \in A$	$V_k \ni v_k^t : O \rightarrow \mathbb{R}$
v_k	Claimed valuation function of agent $k \in A$	$V_k \ni v_k : O \rightarrow \mathbb{R}$
p_k	Payment function for agent $k \in A$	$p_k : V \rightarrow \mathbb{R}$
u_k	Utility function of agent $k \in A$	$u_k(v_k, v_{-k}) = v_k^t(f(v)) + p_k(v)$
v_{-k}	Vector of claimed valuation functions except v_k, $k \in A$	$v_{-k} = (v_1, \dots, v_{k-1}, v_{k+1}, \dots, v_{\|A\|})$
v	Vector of claimed valuation functions	$v = (v_1, \dots, v_{\|A\|})$ $v = (v_k, v_{-k}), k \in A$

$v_{-k} = (v_1, \dots, v_{k-1}, v_{k+1}, \dots, v_{|A|})$. For the sake of notational convenience, we will use v and (v_k, v_{-k}) interchangeably.

The *mechanism* itself is designed and controlled by a central planner. It is a pair (f, p), composed of a *social choice function* $f : V \rightarrow O$ and a vector of *payment functions* $p := (p_1, \dots, p_{|A|})$, with $p_k : V \rightarrow \mathbb{R}$ for all $k \in A$. The mechanism (f, p) is said to *implement* the social choice function f. It is *efficient*, if f maximizes social welfare, i.e., the sum of the valuation functions of all agents (see, e.g., Heydenreich et al. 2008; Mitra 2002, 2001). As described in Sect. 1.1.2, in the context of scheduling problems, the social choice function is an algorithm that determines a feasible schedule based on the valuation functions reported to the mechanism. It is also referred to as the *scheduling rule*, *allocation rule*, or *allocation function*. As the global objective function of a scheduling problem may be *non-utilitarian*, i.e., differ from aiming to maximize the sum of the valuation functions of all agents, we will use the term efficiency in a broader sense by referring to a mechanism as efficient whenever it optimizes the global optimality criterion of the scheduling problem. By controlling the allocation rule and the payment functions, the central planner can design mechanisms with different features.

Each agent $k \in A$ selfishly aims to maximize the *utility function* $u_k : V \to \mathbb{R}$, which is assumed to be *quasi-linear*, i.e., corresponds to the sum of the agent's valuation of the schedule (determined by the allocation rule) and the (potentially negative) corresponding payment from the mechanism, $u_k(v_k, v_{-k}) := v_k^t(f(v_k, v_{-k})) + p_k(v_k, v_{-k})$. Sometimes, it is reasonable to focus on *individually rational* mechanisms (also referred to as *voluntary participation* mechanisms, see Auletta et al. 2004), that assume (or feature) the utilities of each agent to always be nonnegative (see, for instance, Nisan 2007; Hoeksma and Uetz 2013).

1.1.4 Randomized mechanisms and publicly known distributions

All of the above definitions consider a deterministic problem setting and assume that the agents have no information at all about the private information of the other agents. Unless stated otherwise, these will also be our standard assumptions throughout the remainder of this paper. The literature, however, also considers modified settings. Most important, one can assume the allocation rule to be non-deterministic, i.e., let the scheduling algorithm's logic employ some degree of randomness, or consider randomized payments. A resulting mechanism is then referred to as a *randomized mechanism* (Nisan and Ronen 1999, 2001). Moreover, it may sometimes be appropriate to assume that there exists some commonly known probability distribution over the private information of each player (Nisan 2007). In both modified settings, agents are usually assumed to maximize expected utilities. The definitions of the standard setting carry over in a straightforward manner.

1.2 Article overview

The remainder of this article is structured as follows. In Sect. 2, we present an overview of problem categories and problem features that characterize machine scheduling settings in the algorithmic mechanism design literature. This will lay the foundation for our extension of the classification scheme of Graham et al. (1979) in Sect. 3 and allow a structured overview of the literature in Sect. 4. The article closes with a conclusion and an illustration of research challenges that can be identified based on the prior classification of the literature in Sect. 5.

2 Review of problem categories and features

In addition to the classical problem categories of the classification scheme of Graham et al. (1979) and its extensions, the algorithmic mechanism design literature for machine scheduling problems (as restricted in Sect. 1) can be structured based on multiple categories, which we will present in the following sections, where we will also discuss additional problem features that we have not yet introduced.

2.1 Categories, risk attitude, and private information of agents

With respect to categories of agents, there exist two streams of publications. The first group of articles, which follows the seminal work of Nisan and Ronen (1999, 2001), presumes that solely the machines are selfish agents (typically referred to as *machine agents*), i.e., $A = M$. Machine agents usually aim for small loads. Similarly, the second stream of publications assumes that only the jobs are selfish agents (*job agents*), i.e., $A = J$, mostly aiming at small completion times. Prominent examples of the latter stream are Suijs (1996) or Angel et al. (2006). The literature on job agents that are to be scheduled on a single machine sometimes analyzes the *independence of irrelevant alternatives* (IIA) property of allocation rules (see, e.g., Heydenreich et al. 2008). It is satisfied if the relative order of any two jobs on the machine is independent of the committed types of all other jobs.

Of course, one may also think of more general settings, where both (a subset of) jobs and (a subset of) machines represent selfish players of the considered game. Even more general, there might be "owners" of multiple jobs or machines that act as single agents. However, to the best of the authors' knowledge, this setting has not yet been considered in the noncooperative literature.

Concerning the *risk attitude*, the vast majority of research articles assumes the agents to be risk neutral. Exceptions are Kovalyov and Pesch (2014) and Kovalyov et al. (2016), where job agents are assumed to be "fully" risk averse.

Mechanism design settings can also be classified with respect to the knowledge of agents about the private information of the other agents. The standard case is to assume that agents have no information at all about the other's private information. Sometimes, however, one assumes that there exists some commonly known distribution over the private information of each agent (see Sect. 1.1.4) or that there are other publicly known restrictions on the private information of each agent. An example for the latter case is to restrict privately known speed factors of machine agents to be natural numbers that are bounded from above by a publicly known constant (Auletta et al. 2004).

2.2 Truthfulness, VCG mechanisms, and approximability

As indicated in Sects. 1.1.2 and 1.1.3, agents selfishly aim to maximize their (expected) utilities and may therefore lie about their true valuation functions. To overcome this problem, the central planner may want to design the mechanism such that agents behave truthfully. The literature considers different concepts of truthfulness. We will briefly outline the concepts that are relevant for this article in this section.

A mechanism is *(dominant strategy) incentive compatible* or *truthful* (Nisan 2007) if it guarantees that reporting the true valuation function maximizes the utility function of a rationally acting agent for all possible vectors of claimed valuation functions of the other agents, i.e., if $u_k(v_k^t, v_{-k}) \geq u_k(v_k, v_{-k})$ for all $k \in A$, all $v_k \in V_k$, and all $v_{-k} \in V_{-k}$.

In case of randomized mechanisms, articles usually apply an adapted notion of truthfulness, referred to as *truthfulness in expectation*. Formally, let $E(u_k(v))$ denote the expected value of the utility function of agent $k \in A$ over the randomization of the mechanism. A mechanism is truthful in expectation if $E(u_k(v_k^t, v_{-k})) \geq E(u_k(v_k, v_{-k}))$, for all $k \in A$, $v_k \in V_k$, and $v_{-k} \in V_{-k}$. Alternatively, one may slightly deviate from our definition in Sect. 1.1.4 and define a randomized mechanism to allow distributions over deterministic mechanisms. Then, a randomized mechanism is defined to be *universally truthful* if every deterministic mechanism in the support is dominant strategy incentive compatible (Nisan 2007).

Similarly, when considering the case of publicly known probability distributions over the type spaces of agents (that we will denote by Φ_k for agent $k \in A$) as described in Sect. 1.1.4, one can apply a weaker notion of truthfulness, referred to as *Bayes–Nash incentive compatibility* (see, for example, Duives et al. 2015; Heydenreich et al. 2008; Hoeksma and Uetz 2013). Here, for each agent, telling the truth must be (weakly) dominant *in expectation* over the publicly known distributions over the type spaces of the other agents.

One of the most important general results in the field of mechanism design is the *Vickrey–Clarke–Groves mechanism* (VCG mechanism), which was suggested by Vickrey (1961) and generalized by Clarke (1971) and Groves (1973). A mechanism is called a VCG mechanism, if the social choice function maximizes social welfare and if the payment functions $p_k(v)$, $k \in A$, are given by

$$p_k(v) = h_k(v_{-k}) + \sum_{l=1, l \neq k}^{n} v_l(f(v)),$$

where $h_k(v_{-k}) : V_{-k} \to \mathbb{R}$. Note that h_k, $k \in A$, is independent of the valuation function $v_k \in V_k$ reported by agent k. This general definition of the payment functions is referred to as the *Groves payments*. The *Clarke pivot rule* specifies $h_k(v_{-k}) = -\max_{o \in \hat{O}} \sum_{l=1, l \neq k}^{n} v_l(o), k \in A$, where $\hat{O} \subseteq O$ is the set of all feasible schedules that the considered scheduling algorithm may compute. The resulting payment functions have specific desirable properties. They are, for example, non-positive, so that the agents never receive payments from the mechanism. Furthermore, a corresponding mechanism is individually rational if $v_k(o) \geq 0$ for all $k \in A$ and all $o \in \hat{O}$. The concept of VCG mechanisms was further generalized by Roberts (1979) to social choice functions that belong to the set of so-called *affine maximizers*.

A VCG mechanism is (dominant strategy) incentive compatible, but a major drawback is the need to find optimal solutions to the underlying problem of maximizing social welfare, which may be NP-hard (see, for instance, Nisan 2007). Hence, in the context of scheduling problems, VCG mechanisms are oftentimes not appropriate even if the objective function of the specific scheduling problem corresponds to maximizing social welfare. Many researchers are therefore actively trying to explore the boundary between truthfulness (in its most general sense) and computational complexity for specific scheduling problems (see also Sect. 5). It is especially desirable to derive truthful mechanisms with polynomial time computable allocation and payment functions that feature good approximation factors for specific settings, and to determine strong dual bounds on approximation factors for truthful scheduling under non-utilitarian objectives without restricting the analysis to polynomial time allocation functions. If one aims to design a concrete polynomial time truthful mechanism for some NP-hard scheduling domain, one must make use of theoretical results that are related to incentive compatibility and that are suitable for heuristic algorithms. For example, one may have to assure that certain *monotonicity* conditions are fulfilled by the allocation rule in order to guarantee incentive compatibility (see, e.g., Heydenreich et al. 2007; Lavi and Swamy 2009, for more details).

2.3 Models of execution and constraints on committed data

When agents possess private information on the processing times of jobs, the literature distinguishes between different *models of execution*. These models differ in the length of the actual time slots that are reserved on the machines once the schedule that has been determined by the scheduling algorithm is implemented. There are two widely used models of execution (see, e.g., Christodoulou et al. 2007a). The *strong model of execution* assumes that the schedules which are implemented based on the computations of the scheduling algorithms always apply the true processing times of the jobs, no matter which value has been committed by the agents. In contrast, in the *weak model of execution*, the implemented schedules use the reported processing times. A third model of execution is used by Koutsoupias (2014). Here, the implemented processing times are defined by the maximum of the reported and the true processing times. We refer to this model of execution as the *maximum model of execution*. The weak and maximum models of execution are especially relevant in applications where the processing of all jobs can actually be observed by the agents. If the true processing times were applied in this case, a lie of an agent who commits a processing time that is larger than its true value would be observable by the other agents, which might not be desirable in the considered application (Koutsoupias 2014). Similarly, there may be applications where the strong model of execution is most suitable, for example, when machine idle times are very costly for their owner or not allowed at all.

Additionally, one may want to *constrain* the data that the agents are allowed to report to the mechanism without publicly revealing explicit data of their private information. For example, in case of private processing times (release dates), it may be reasonable to restrict the agents to commit processing times (release dates) that are bounded from below by their true values (see, e.g., Angel et al. 2012; Christodoulou et al. 2007a).

2.4 Characteristics of payment schemes

There exist many applications, where mechanisms may be restricted to *not include payments* for compensation purposes, i.e., where $p_k(v) = 0$ must hold for all $k \in A$ and $v \in V$ (see, for example, Koutsoupias 2014). Very broadly speaking, this "constraint can arise from ethical and/or institutional considerations: Many political decisions must be made without monetary transfers; organ donations can be arranged by 'trade' involving multiple needy patients and their relatives, yet monetary compensation is illegal" (Schummer and Vohra 2007). In a scheduling context, a setting with payments "is easily challenged when it comes to computational settings. In particular in internet domains payments are notoriously difficult to implement, mainly due to security and banking issues" (Procaccia and Tennenholtz 2009).

Other applications strive for mechanisms that have no surplus or deficit of monetary payments that cannot be redistributed among the agents (Suijs 1996), i.e., where $\sum_{k \in A} p_k(v) = 0$ for all $v \in V$. These payment schemes are usually referred to as *budget-balanced*.

Some articles aim to design Bayes–Nash or dominant strategy incentive compatible mechanisms for scheduling problems that *minimize the sum of the (expected) payments* (see, for instance, Duives et al. 2015; Heydenreich et al. 2008; Hoeksma and Uetz 2013). Mechanisms of this type are usually referred to as *optimal mechanisms* (see also Hartline and Karlin 2007). Usually, these articles focus on individually rational mechanisms or some approximation guarantee of the scheduling algorithms (given truthful commitments of the agents).

2.5 Other problem categories and features

There exist some mechanism design-related problem features that do not fall into the categories of the above sections. In the following, we list the features that are relevant for this article.

A mechanism is called *anonymous* if, whenever two agents switch all of their properties, these two agents also switch positions in the resulting schedule (see, for example, Ashlagi et al. 2012).

In a mechanism with *verification*, the calculation of the payments depends on the results of the execution of the schedule (see, e.g., Nisan and Ronen 1999, 2001). If, for example, the processing time of a job is part of the private information of an agent, additional information on the true processing time becomes available after the execution of the schedule, which depends on the model of execution.

Next, a mechanism is called *envy-free*, if no agent is able to improve her utility function value by switching both, the position in the schedule and the realized payments, with another agent (see, for instance, Kayı and Ramaekers 2010, 2015).

A mechanism is *task-independent* if its allocation function decides on the allocation of each job separately, i.e., without considering the characterizing parameters of the other jobs (see, for instance, Christodoulou et al. 2010; Dobzinski and Sundararajan 2008).

When considering a mechanism design setting with machine agents, a mechanism is called *locally decisive*, if each agent can enforce her allocation by reporting very low or high values but cannot determine how the remaining jobs are allocated among the other agents (see, e.g., Christodoulou et al. 2008).

3 Classification scheme

We are now ready to present our extension of the classification scheme of Graham et al. (1979). The resulting scheme is extensible, i.e., it allows for including more features when needed.

3.1 Review of selected elements of Graham et al. (1979)

We will first review parts of the notation introduced by Graham et al. (1979). With respect to the machine environment α, they define:

- **Machine environment**, $\alpha_1 \in \{\circ, P, Q, R, \ldots\}$

$\circ$	Single machine, i.e., $m = 1$.
P	Identical (parallel) machines.
Q	Uniform (parallel) machines.
R	Unrelated (parallel) machines.

- **Number of machines**, $\alpha_2 \in \{\circ, \mathbb{N}\}$

$\circ$	m is variable.
positive integer m	There exists a constant number m of machines.

Regarding the job characteristics β, we will only make use of a few elements introduced by Graham et al. (1979):

- **Release dates**, $\beta_4 \in \{\circ, r_j\}$

$\circ$	No release dates are specified.
r_j	Release dates per job are specified.

- **Processing times**, $\beta_6 \in \{\circ, t_j = 1, \ldots\}$

$\circ$	Processing times are arbitrary.
$t_j = 1$	Each job has unit processing time.

Furthermore, we will make use of the following element (see, for example, Leung and Li 2008):

- **Eligibility constraints**, $\beta_7 \in \{\circ, M_j\}$

$\circ$	No eligibility constraints are specified.
M_j	Each job $j \in J$ is associated with a set of eligible machines $M_j \subseteq M$ on which it can be processed.

Finally, based on Graham et al. (1979) and with respect to the (global) optimality criterion γ, i.e., the objective function that the scheduling algorithm aims to optimize based on the publicly known parameters and the values committed by the agents, we define:

- **Global optimality criterion**, $\gamma \in \{C_{max}, \sum C_j, \sum w_j C_j, \max w_j C_j, \sum f_j(C_j), \sum f(C_j), \sum w_j f(C_j), \sum w_j U_j, \sum f(L_i), \sum L_i, \|L\|_p, \max\min L_i, \ldots\}$:

C_{max}	Minimize the makespan, i.e., the maximum of the completion times.
$\sum C_j$	Minimize the sum of completion times.
$\sum w_j C_j$	Minimize the weighted sum of completion times.
$\max w_j C_j$	Minimize the maximum weighted completion time.
$\sum f_j(C_j)$	Minimize the sum of functions f_j of the completion times of jobs $j \in J$.
$\sum f(C_j)$	Minimize the sum of a function f of the completion times.
$\sum w_j f(C_j)$	Minimize the weighted sum of a function f of the completion times.
$\sum w_j U_j$	Minimize the total weight of late jobs.
$\sum f(L_i)$	Minimize the sum of a function f of the load of the machines.
$\sum L_i$	Minimize the sum of the loads of the machines.
$\|L\|_p$	Minimize the l_p norm of the vector of machine loads L.
$\max\min L_i$	Maximize the minimum load over all machines.

3.2 Including mechanism design settings for machine scheduling problems

We can now propose additional notation that allows to include mechanism design settings. As described in Sect. 2.1, the existing literature can be divided into two groups of articles that either presume the existence of machine agents or job agents. We therefore augment the first two fields of the classification scheme of Graham et al. (1979), that represent the machine environment α and the job characteristics β, with additional elements $\hat{\alpha}_l$ and $\hat{\beta}_l$, $l = 1, 2, \ldots$, respectively. Here, the index l refers to the l-th element that refers to mechanism design (indicated by the hat operator) characteristics in the specific field.

First, in order to be able to indicate the risk attitude of the agents, we define the elements $\hat{\alpha}_1$ for machine agents and $\hat{\beta}_1$ for job agents:

- **Risk attitude of agents**, $\hat{\alpha}_1, \hat{\beta}_1 \in \{\circ, averse, seeking, \ldots\}$

$\circ$	No agents at all (no mechanism design setting) or all agents are risk neutral.
$averse$	All agents are risk averse.
$seeking$	All agents are risk seeking.

The elements $\hat{\alpha}_2$ and $\hat{\beta}_2$ specify the set of parameters that are private information of the agents and indicate whether the other agents have some common knowledge about this private information.

- **Private information of machine agents**, $\hat{\alpha}_2 \in \{\circ, priv_\rho\{\ldots\}, \ldots\}$

$\circ$	No machine agents.
$priv_\rho\{\ldots\}$	Each element of the set $\{\ldots\}$ refers to an entity of private information of the machine agents. A superscript τ for some element indicates additional restrictions or assumptions on the agents' commitments or their implementation *that do not publicly reveal explicit data* of the private information. Multiple entries in the superscript are separated by commas.

  - s_i^τ: Machine agent $i \in M$ has private information on the speed factor s_i. Potential entries in superscript τ:
    - …
  - t_{ij}^τ: Machine agent $i \in M$ has private information on the processing times t_{ij} for all jobs $j \in J$. Potential entries in superscript τ:
    - max: The maximum model of execution is applied.
    - …
  - …

  The subscript ρ indicates additional restrictions or assumptions (on each piece of private information that can be committed by the machine agents) *that are based on publicly known parameters*. Multiple entries in the subscript are separated by commas. Potential entries in subscript ρ:

  - Φ: A distribution of each machine agent's private information is publicly known.
  - d_ω: The values that can be reported by the machine agents are restricted to be elements of publicly known discrete sets. Each element of the subscript ω indicates additional restrictions on these sets:
    - f: The sets are finite for all machine agents.
    - $=_i$: The sets are identical for all machine agents.
    - $=_j$: The sets are identical for all jobs (if relevant).
    - div: The private information is *divisible*, i.e., it belongs to a set $C = \{c_1, c_2, \ldots\}$, where c_{i+1} is a multiple of c_i $\forall i$.
    - c-div: The private information is *c-divisible*, i.e., it is a power of a given positive constant c.
    - …
  - c_ω: The values that can be reported by the machine agents are restricted to be elements of publicly known bounded and continuous sets. Each element of the subscript ω indicates additional restrictions on these sets:
    - $=_i$: The sets are identical for all machine agents.
    - …
  - …

- **Private information of job agents**, $\hat{\beta}_2 \in \{\circ, priv_\rho\{\ldots\}, \ldots\}$

$\circ$	No job agents.
$priv_\rho\{\ldots\}$	Each element of the set $\{\ldots\}$ refers to an entity of private information of the job agents. A superscript τ for some element indicates additional restrictions or assumptions on the agents' commitments or their implementation *that do not publicly reveal explicit data* of the private information. Multiple entries in the superscript are separated by commas.

- r_j^τ: Job agent $j \in J$ has private information on its release date r_j. Potential entries in superscript τ:
  - $\geq$: The release date committed by job agent $j \in J$ is bounded from below by the true release date.
  - …
- d_j^τ: Job agent $j \in J$ has private information on its due date or deadline d_j. Potential entries in superscript τ:
  - …
- w_j^τ: Job agent $j \in J$ has private information on its weight w_j. Potential entries in superscript τ:
  - …
- f_j^τ: Job agent $j \in J$ has private information on the function f_j that maps every possible completion time of its job to a real value. Potential entries in superscript τ:
  - …
- t_j^τ: Job agent $j \in J$ has private information on its processing time t_j. Potential entries in superscript τ:
  - $strong$: The strong model of execution is applied.
  - $weak$: The weak model of execution is applied.
  - $\geq$: The processing times committed by job agent $j \in J$ are bounded from below by the true processing times.
  - …
- t_{ij}^τ: Job agent $j \in J$ has private information on its processing times t_{ij} that may differ among machines $i \in M$. The potential entries of the superscript τ are defined as above.
- …

The subscript ρ indicates additional restrictions or assumptions (on each piece of private information that can be committed by the job agents) *that are based on publicly known parameters*. Multiple entries in the subscript are separated by commas. Potential entries in subscript ρ:

- Φ: A distribution of each job agent's private information is publicly known.
- d_ω: The values that can be reported by the job agents are restricted to be elements of publicly known discrete sets. Each element of the subscript ω indicates additional restrictions on these sets:
  - f: The sets are finite for all job agents.
  - …
- …

Finally, we define elements $\hat{\alpha}_3$ and $\hat{\beta}_3$ that represent the (true) valuation functions of the agents, i.e., their "local" objective functions related to the scheduling problem.

- **Objective of agents**, $\hat{\alpha}_3 \in \{L_i, \dots\}$, $\hat{\beta}_3 \in \{C_j, U_j \dots\}$

$\hat{\alpha}_3 = L_i$	Each machine agent $i \in M$ aims to minimize its load L_i.
$\hat{\beta}_3 = C_j$	Each job agent $j \in J$ aims to minimize its completion time C_j.
$\hat{\beta}_3 = U_j$	Each job agent $j \in J$ aims to complete before or at d_j, i.e., to minimize the unit penalty function U_j.

As mentioned above, our classification scheme is extensible. This is indicated by dots in the above notation, which allow for including new problem settings for existing categories, for example, when considering new entities of private information of agents. Furthermore, when considering more general categories of agents or similar problem generalizations or extensions, one can include new symbols to represent those settings.

3.3 Examples

We will now illustrate the above classification scheme by presenting two examples.

$P||C_{\max}$: We are given an arbitrary number of m parallel identical machines and a set J of n jobs. The processing time t_j of any job $j \in J$ is independent of the machines. Each job can be processed by at most one machine at a time, and each machine is capable of processing at most one job at a time. The objective is to assign each job to exactly one machine and find non-preemptive sequences of the resulting subsets of jobs of each machine, so that the makespan is minimized. There is no private information; a mechanism design setting is not considered.

$P|priv\{t_j^{\text{strong},\geq}\}, C_j|C_{\max}$: The setting is in analogy to $P||C_{\max}$, but we now consider a mechanism design setting with job agents who aim to minimize their completion times. The processing time of each job agent is private information. The processing times committed to the mechanism are bounded from below by their true values. The strong model of execution is applied.

4 Literature overview

Based on the classification scheme of Sect. 3, we can now present a structured overview of the relevant literature. We will do so by presenting three tables that refer to articles that consider problem settings with job agents (Table 3), machine agents and unrelated machines (Table 4), and machine agents and uniform machines (Table 5).

We will make use of some additional definitions regarding approximability results that we briefly introduce before describing the tables in detail. For a given constant $c \in \mathbb{R}$, a polynomial algorithm is called a *c-approximation algorithm* if (in case of a minimization problem) the objective function value determined by the algorithm is bounded from above by c times the optimal objective function value. A polynomial algorithm with an additional arbitrary positive input parameter ε that guarantees an objective function value which is bounded from above by $1 + \varepsilon$ times the optimal objective function value is called a *polynomial-time approximation scheme* (PTAS). If the runtime of the algorithm is polynomial in the size of the problem, denoted by q, and in $1/\varepsilon$, the algorithm is called *fully polynomial-time approximation scheme* (FPTAS). Another variant of a PTAS is a *quasi-polynomial-time approximation scheme* (QPTAS). A QPTAS has time complexity $q^{polylog(q)}$.

Within the tables, each article, identified by its authors and publishing year, is classified according to the extended classification scheme presented in Sect. 3. Furthermore, the tables highlight the main contributions of each article (as explicitly stated by the authors) by specifying whether it focuses on selected properties or restricts the analysis to specific settings. These selected properties and restrictions slightly differ among the tables because the literature related to each table usually takes a fairly specific perspective on mechanism design settings for machine scheduling problems.

With respect to the characteristics of the payment scheme (Paym.), we indicate whether an article considers nonzero payments at all ($\exists$) and whether the presented payment scheme is budget-balanced (BB). Furthermore, we present information on the question of whether or not the mechanisms presented in an article are individually rational (IR). With regard to the considered mechanisms (Mech.), we indicate if they are randomized (rand.) or deterministic (det.). The column "Opt. Me." indicates if the authors consider the design of optimal mechanisms. Similarly, the focus on different concepts of truthfulness is illustrated in the column "Truthfuln.", where we specify whether the authors present results on or restrict their analysis to dominant strategy truthfulness (DS), focus on universally truthful mechanisms (U), consider truthfulness in expectation (IE), or examine Bayes–Nash incentive compatibility (BN).

Furthermore, the tables contain information on the approximability of the considered settings. Most important, we indicate results concerning dual and primal bounds (DB/PB) on approximation factors in the "Approx. Quality" column. Note that, with respect to the dual bounds, the studies under consideration do not restrict to what polynomial time mechanisms can achieve, i.e., the dual bounds even hold for exponential time allocation (and payment) functions (see Sect. 2.2). We additionally indicate whether a primal bound presented in some article is based on a polynomial time allocation function and whether a PTAS, FPTAS, or QPTAS is considered (column "poly."). If, in this context, the allocation function is a polynomial time algorithm, but the payments are not necessarily computable in polynomial time (and this is also highlighted by the respective authors), this is specified in the last column of a table.

In the last column of the table, we present highlights of the articles as well as additional properties (Table 3) or additional restrictions and assumptions on the considered setting (Tables 4 and 5) that are not covered by the preceding columns. In this context, note that some articles consider relaxations of the original scheduling problems, i.e., by allowing a job to be split into arbitrary fractions among the machines. This is usually referred to as *fractional scheduling*.

Table 3 Overview—Job agents

Authors	Classification	Paym.		IR	Mech.		Opt.	Truthfuln.		Approx. quality			Properties/highlights
		∃	BB		rand.	det.	Me.	DS/U	BN/IE	DB	PB	poly.	
Suijs (1996)	$1\|priv\{w_j\}, C_j\|\sum w_jC_j$	●	+	-	○	●		+	+		1	+	
Hamers et al. (1999)	$P\|t_j=1, priv\{w_j\}, C_j\|\sum w_jC_j$	●	+		○	●		+	+		1	+	
Mitra (2001)	$1\|t_j=1, priv\{f_j\}, C_j\|\sum f_j(C_j)$	●	+*	+*	○	●		+*	+*		1*		
Mitra (2002)	$1\|priv\{w_j\}, C_j\|\sum w_j f(C_j)$	●	+*		○	●		+*	+*		1*		
Hain and Mitra (2004)	$1\|priv\{t_j^{strong}\}, C_j\|\sum f(C_j)$	●	+*		○	●		+*	+*		1*		
Angel et al. (2005, 2006)	$P\|priv\{t_j^{strong,\geq}\}, C_j\|C_{max}$	○	+		●	○			+		$2-\frac{1}{m+1}(\frac{5}{3}+\frac{1}{3m})$	+	
Mitra (2005)	$P\|t_j=1, priv\{f_j\}, C_j\|\sum f_j(C_j)$	●	+*		○	●		+*	+*		1*		
Christodoulou et al. (2007a)	$P\|priv\{t_j^{strong,\geq}\}, C_j\|C_{max}$	○	+		●	○			+	$\frac{3}{2}-\frac{1}{2m}$			
	$P\|priv\{t_j^{strong,\geq}\}, C_j\|C_{max}$	○	+		○	●		+	+	$2-\frac{1}{m}$			
	$P\|priv\{t_j^{weak,\geq}\}, C_j\|C_{max}$	○	+		○	●		+	+	$\frac{7}{6}$*	$\frac{4}{3}-\frac{1}{3m}$	-	
	$P2\|priv\{t_j^{weak,\geq}\}, C_j\|C_{max}$	○	+		○	●		+	+	$1+\frac{\sqrt{105}-9}{12}$			
Heydenreich et al. (2008)	$1\|priv_{\Phi}\{w_j\}, C_j\|\sum w_jC_j$	●		+	○	●	+*		+*		1*	+	Closed formula for payments*
	$1\|priv_{\Phi,d_f}\{w_j\}, C_j\|\sum w_jC_j$	●		+	○	●	+*		+*		1*	+	Closed formula for payments*
	$1\|priv_{\Phi,d_f}\{w_j, t_j^{weak,\geq}\}, C_j\|\sum w_jC_j$	●		+	○	●	+		+*	>1			Opt. mech. does not satisfy IIA
Angel et al. (2009)	$Q\|priv\{t_j^{weak,\geq}\}, C_j\|C_{max}$	○	+		●	○			+		$r(1+\varepsilon)$ [a]	+	
	$Q\|priv\{t_j^{weak,\geq}\}, C_j\|C_{max}$	●			●	○			+		$1+\varepsilon$	+	

Table 3 continued

Authors	Classification	Paym.		IR	Mech.		Opt.	Truthfuln.		Approx. quality			Properties/highlights
		∃	BB		rand.	det.	Me.	DS/U	BN/IE	DB	PB	poly.	
Angel et al. (2010, 2012)	$P\|priv\{t_j^{strong,\geq}\}, C_j\|C_{max}$	○	+		●	○			+		1.5	-	
	$P\|priv\{t_j^{strong,\geq}\}, C_j\|C_{max}$	○	+		●	○			+		$\frac{11}{6}+\frac{1}{3m}$	+	
	$Pm\|priv\{t_j^{strong,\geq}\}, C_j\|C_{max}$	○	+		●	○			+		$1.5+\varepsilon$	+	
	$P\|r_j, priv\{t_j^{weak,\geq}, r_j^{\geq}\}, C_j\|C_{max}$	○	+		●	○			+		1.5	-	
	$R\|priv\{t_{ij}^{weak,\geq}\}, C_j\|C_{max}$	○	+		●	○			+		1.5	-	
Kayı and Ramaekers (2010, 2015)	$1\|t_j=1, priv\{w_j\}, C_j\|\sum w_jC_j$	●	+		○	●		+	+		1		Envy-free, anonymous
Hashimoto and Saitoh (2012)	$1\|t_j=1, priv\{w_j\}, C_j\|\sum w_jC_j$	●	+		○	●		+	+		1		Anonymous
Hoeksma and Uetz (2013)	$1\|priv_{\Phi,d_f}\{w_j, t_j^{weak,\geq}\}, C_j\|\sum w_jC_j$	●		+	●	○	+		+		1	+	
Kovalyov and Pesch (2014)	$1\|averse, priv\{f_j\}, C_j\|\sum f_j(C_j)$	●	+		●	○			+				
Duives et al. (2015)	$1\|priv_{\Phi,d_f}\{w_j\}, C_j\|\sum w_jC_j$	●		+	○	●	+	+	+	>1*	1*	+	Closed formula for payments
	$1\|priv_{\Phi,d_f}\{w_j, t_j^{weak,\geq}\}, C_j\|\sum w_jC_j$	●		+	○	●	+		+*	>1			Opt. mech. does not satisfy IIA
Kovalyov et al. (2016)	$P\|averse, priv\{w_j, d_j\}, U_j\|\sum w_jU_j$	●	+		●	○			+				Computational analysis
De and Mitra (2017)	$1\|priv\{w_j\}, C_j\|\max w_jC_j$	●	+		○	●		+	+		1		
	$1\|priv\{w_j, t_j^{weak}\}, C_j\|\max w_jC_j$	●	+		○	●		+	+		1		
Kress et al. (2017)	$1\|priv\{w_j, d_j\}, U_j\|\sum w_jU_j$	●			○	●		+	+		$n-1$	+	
	$P\|priv\{w_j, d_j\}, U_j\|\sum w_jU_j$	●			○	●		+	+				General characterization of DS truthfulness

[a] with $s_1 \leq s_2 \leq \ldots \leq s_m$, and $r := s_m/s_1$

Table 4 Overview—unrelated machine agents

Authors	Classification	Paym.	Mech.		Truthfuln.		Approx. quality			Additional restrictions or assumptions/highlights
		∃	rand.	det.	DS/U	BN/IE	DB	PB	poly.	
Nisan and Ronen (1999, 2001)	Rm, $priv_{c=_i}\{t_{ij}\}$, $L_i\|\|C_{max}$	●	○	●	+	+		$1+\varepsilon$	PTAS	Verification
	R, $priv\{t_{ij}\}$, $L_i\|\|C_{max}$	●	○	●	+	+	2	m	+	
	$R2$, $priv\{t_{ij}\}$, $L_i\|\|C_{max}$	●	●	○		+		1.75	+	
Christodoulou et al. (2007c, 2009b)	R, $priv\{t_{ij}\}$, $L_i\|\|C_{max}$	●	○	●	+	+	$1+\sqrt{2}$			$m \geq 3$
Christodoulou et al. (2007b, 2010)	R, $priv\{t_{ij}\}$, $L_i\|\|C_{max}$	●	○	●	+	+	$2-\frac{1}{m}$	$\frac{m+1}{2}$	+	Fractional
	R, $priv\{t_{ij}\}$, $L_i\|\|C_{max}$	●	○	●	+	+	$\frac{m+1}{2}$	$\frac{m+1}{2}$	+	Fractional, task-independent
Koutsoupias and Vidali (2007, 2013)	R, $priv\{t_{ij}\}$, $L_i\|\|C_{max}$	●	○	●	+	+	≈ 2.618			$m \geq 4$
Lavi and Swamy (2007, 2009)	R, $priv_{d_f,=_i}\{t_{ij}\}$, $L_i\|\|C_{max}$	●	●	○		+		3	+	General technique, payments not necessarily polynomial time computable
	R, $priv_{d_f,=_i,=_j}\{t_{ij}\}$, $L_i\|\|C_{max}$	●	○	●	+	+	1.1	2	+	
Mu'alem and Schapira (2007, 2017)	R, $priv\{t_{ij}\}$, $L_i\|\|C_{max}$	●	●	○	+	+	$2-\frac{1}{m}$	$0.875m$	+	
	R, $priv_{\Phi_{d_f}}\{t_{ij}\}$, $L_i\|\|C_{max}$	●	○	●	+	+	1.25			
Christodoulou et al. (2008)	$R2$, $priv\{t_{ij}\}$, $L_i\|\|C_{max}$	●	○	●	+	+	2			Locally decisive, negative processing times allowed, characterization of truthful mechanisms
Dobzinski and Sundararajan (2008)	$R2$, $priv\{t_{ij}\}$, $L_i\|\|C_{max}$	●	●	○	+					Every universally truthful mechanism that yields a finite approximation factor is such that all mechanisms in its support are task-independent
Lu and Yu (2008a)	R, $priv\{t_{ij}\}$, $L_i\|\|C_{max}$	●	●	○	+	+		$0.8368m$	+	

Table 4 continued

Authors	Classification	Paym.	Mech.		Truthfuln.		Approx. quality			Additional restrictions or assumptions/highlights
		∃	rand.	det.	DS/U	BN/IE	DB	PB	poly.	
Lu and Yu (2008b)	$R2, priv\{t_{ij}\}, L_i\|\|C_{max}$	●	●	○	+	+	$\frac{11}{7}$	1.5963	+	Task-independent
	$R, priv\{t_{ij}\}, L_i\|\|C_{max}$	●	●	○		+	$\frac{m+1}{2}$	$\frac{m+5}{2}$	+	Task-independent
Ashlagi et al. (2009, 2012)	$R, priv\{t_{ij}\}, L_i\|\|C_{max}$	●	○	●	+	+	m			Anonymous
	$R, priv\{t_{ij}\}, L_i\|\|\|L\|_p$	●	○	●	+	+	$m^{1-\frac{1}{p}}$			Anonymous
	$R, priv\{t_{ij}\}, L_i\|\|\sum C_j$	●	○	●	+	+	m			Anonymous
Christodoulou and Kovács (2011)	$R, priv\{t_{ij}\}, L_i\|\|C_{max}$	●	○	●	+	+				Characterization of anonymous envy-free mechanisms with $n = 2$
Koutsoupias (2011, 2014)	$R, priv\{t_{ij}^{max}\}, L_i\|\|C_{max}$	○	●	○		+	$\frac{m+1}{2}$	$\frac{m+1}{2}$	+	$n = 1$
	$R, priv\{t_{ij}^{max}\}, L_i\|\|C_{max}$	○	●	○		+		$\frac{m(m+1)}{2}$	+	
	$R, priv\{t_{ij}^{max}\}, L_i\|\|\sum L_i$	○	●	○		+	$\frac{m+1}{2}$	$\frac{m+1}{2}$	+	
Auletta et al. (2012, 2015)	$R, priv_{d_f}\{t_{ij}\}, L_i\|\|C_{max}$	●	●	○	+*	+*				Various results on approximation quality
Chawla et al. (2013)	$R, priv_{\Phi}\{t_{ij}\}, L_i\|\|C_{max}$	●	○	●	+	+		$\frac{n}{m}$*		
Chen et al. (2013, 2015)	$R2, priv\{t_{ij}\}, L_i\|\|C_{max}$	●	●	○		+		1.58606	+	
Daskalakis and Weinberg (2015)	$R, priv_{\Phi_{d_f}}\{t_{ij}\}, L_i\|\|C_{max}$	●	○	●	+	+		2	+	
Giannakopoulos and Kyropoulou (2015)	$R, priv_{\Phi}\{t_{ij}\}, L_i\|\|C_{max}$	●	○	●	+	+		$\frac{\ln m}{\ln \ln m}$*		

Table 5 Overview—uniform machine agents

Authors	Classification	Paym.	IR	Mech.		Truthfuln.		Approx. Quality			Additional restrictions or assumptions/highlights
		∃		rand.	det.	DS/U	IE	DB	PB	poly.	
Archer and Tardos (2001); Archer (2004)	$Q, priv\{s_i\}, L_i\|\|C_{max}$	●	+	○	●	+	+		1	-	
	$Q, priv\{s_i\}, L_i\|\|C_{max}$	●	+	●	○		+		2	+	
	$Q, priv\{s_i\}, L_i\|\|\sum C_j$	●	+	○	●	+	+		1	+	
	$Q, priv\{s_i\}, L_i\|\|\sum w_j C_j$	●		○	●	+	+	$\frac{2}{\sqrt{3}}$			
Auletta et al. (2004)	$Qm, priv_{d_{f,=_i}}\{s_i\}, L_i\|\|C_{max}$	●	+	○	●	+	+		$2+\varepsilon$	+	
	$Qm, priv_{d_{=_i,div}}\{s_i\}, L_i\|\|C_{max}$	●	+	○	●	+	+		$2+\varepsilon$	+	
	$Qm, priv\{s_i\}, L_i\|\|C_{max}$	●	+	○	●	+	+		$4+\varepsilon$	+	
Ambrosio and Auletta (2005, 2008)	$Q2, priv_{d_{=_i,c-div}}\{s_i\}, L_i\|\|C_{max}$	●		○	●	+*	+*				
	$Q, priv_{d_{=_i,c-div}}\{s_i\}, L_i\|\|C_{max}$	●		○	●	+	+		$2+\varepsilon$	+	Experimental analysis
Andelman et al. (2005, 2007)	$Q, priv\{s_i\}, L_i\|\|C_{max}$	●		○	●	+	+		5	+	
	$Qm, priv\{s_i\}, L_i\|\|C_{max}$	●		○	●	+	+		$1+\varepsilon$	FPTAS	
Kovács (2005)	$Q, priv\{s_i\}, L_i\|\|C_{max}$	●	+	○	●	+	+		3	+	
Kovács (2006, 2009)	$Q, priv\{s_i\}, L_i\|\|C_{max}$	●	+	○	●	+	+		2.8	+	
Lavi (2007)	$Q, priv\{s_i\}, L_i\|\|C_{max}$	●		●	○		+		2	+	
	$Q, priv\{s_i\}, L_i\|\|C_{max}$	●		○	●	+	+		5	+	

Table 5 continued

Authors	Classification	Paym.	IR	Mech.		Truthfuln.		Approx. Quality			Additional restrictions or assumptions/highlights
		∃		rand.	det.	DS/U	IE	DB	PB	poly.	
Dhangwatnotai et al. (2008, 2011)	$Q, priv\{s_i\}, L_i\|\|C_{max}$	●		●	○		+		$1+\varepsilon$	PTAS	
	$Q, priv\{s_i\}, L_i\|\|C_{max}$	●		○	●	+	+		$1+\varepsilon$	QPTAS	
	$Q, priv\{s_i\}, L_i\|\|\|L\|_p$	●		●	○		+		$1+\varepsilon$	PTAS	
	$Q, priv\{s_i\}, L_i\|\|\|L\|_p$	●		○	●	+	+		$1+\varepsilon$	QPTAS	
	$Q, priv\{s_i\}, L_i\|\|\max\min L_i$	●		●	○		+		$1+\varepsilon$	PTAS	
Epstein and van Stee (2008, 2010)	$Qm, priv\{s_i\}, L_i\|\|\max\min L_i$	●		○	●	+	+		$1+\varepsilon$	PTAS, FPTAS	
	$Q, priv\{s_i\}, L_i\|\|\max\min L_i$	●		○	●	+	+		$\min(m, (2+\varepsilon)\frac{s_1}{s_m})$ [a]	+	
Ferrante et al. (2009)	$Q, priv_{c=i}\{s_i\}, L_i\|\|C_{max}$	●	+	○	●	+	+		$c(1+\varepsilon)$ [b]	+	Verification
	$Q, priv_{d=i}\{s_i\}, L_i\|\|C_{max}$	●	+	○	●	+	+		c [b]	+	Verification
Christodoulou and Kovács (2010, 2013)	$Q, priv\{s_i\}, L_i\|\|C_{max}$	●		○	●	+	+		$1+\varepsilon$	PTAS	
Epstein et al. (2013, 2015)	$Q, priv\{s_i\}, L_i\|\|C_{max}$	●		○	●	+	+		$1+\varepsilon$	PTAS	
	$Q, priv\{s_i\}, L_i\|\|\max\min L_i$	●		○	●	+	+		$1+\varepsilon$	PTAS	
	$Q, priv\{s_i\}, L_i\|\|\sum f(L_i)$	●		○	●	+*	+*		$1+\varepsilon$*	PTAS*	
Azar et al. (2015, 2017)	$Q, priv\{s_i\}, L_i\|M_j\|C_{max}$	●		●	○		+		2	+	General technique to design IE truthful mechanisms

[a] with $s_1 \geq \cdots \geq s_m$
[b] $c :=$ approximation quality of an algorithm with certain monotonicity properties

In order to precisely present the information in the tables, we use different symbols to indicate if and how a property or restriction is handled or considered in a given article. First, we use circles to indicate whether a given property is fulfilled or a restriction is considered (●) or not (○). If the authors of an article have proven a given property to be fulfilled or restrict their analysis to a specific setting, we use a "+", if they have proven that a given property is not fulfilled, we use a "-". A blank space indicates that a property or restriction is not explicitly addressed in the corresponding article. If a property is only fulfilled under additional constraints that are not stated elsewhere in the table, we mark these results with a "*".

5 Research challenges and conclusion

In this article, we have provided a classification scheme and a literature overview on direct revelation mechanism design settings in the context of machine scheduling problems. Based on our classification of the literature, we can identify multiple potential directions and challenges for future research that we briefly outline in this section.

First and foremost, as described in Sect. 2.2, it is challenging to better understand the boundary between truthfulness on the one side and computational complexity and approximation on the other side. For example, in case of $R, priv\{t_{ij}\}, L_i||C_{max}$, i.e., a *multi-parameter setting* where machine agents possess more than one piece of private information, Nisan and Ronen (1999, 2001) show that truthfulness excludes optimal solutions to the underlying scheduling problem (even when allowing exponential time allocation and payment functions) by providing a dual bound of 2 on the corresponding approximation factor. This bound can, however, be beaten when allowing randomization (see also Chen et al. 2015; Lu and Yu 2008a, b; Mu'alem and Schapira 2017), which provides evidence for the "power of randomization in algorithmic mechanism design" (Dobzinski and Dughmi 2013). Nisan and Ronen (1999, 2001) furthermore conjecture that no truthful deterministic mechanism can achieve an approximation factor better than m (see Table 4). This so-called *Nisan–Ronen conjecture* has been settled by Ashlagi et al. (2009, 2012) when restricting the analysis to anonymous mechanisms, but it is still open for the general case. A related predominant question, which has especially been studied a single-parameter setting (see Table 5), is whether or not truthful "efficient computation [is] fundamentally less powerful than 'classical' efficient computation" (Dhangwatnotai et al. 2008). Recently, for example, Christodoulou and Kovács (2010, 2013) and Epstein et al. (2013, 2015) have provided the first truthful PTAS for $Q, priv\{s_i\}, L_i||C_{max}$, the existence of which had been a major open question ever since the work of Archer and Tardos (2001), who provided a truthful exponential time deterministic mechanism for minimizing the makespan in this setting.

There also remain plenty of open research questions with respect to classical machine scheduling settings. The existing research presented in Tables 3, 4 and 5, for example, solely focuses on parallel machine settings, while it may also be interesting to analyze dedicated machine environments (flow shop, open shop, or job shop problems) in a game-theoretic context. Furthermore, the literature rarely considers varying job characteristics, as, for example, release dates in offline-settings or precedence relations.

Another interesting avenue for future research is to shift the research focus toward new applications. Most existing research is motivated by the emergence of the internet as a platform for computations (e.g., Nisan and Ronen 2001), for example, when considering the problem of determining an execution sequence for (selfish) computer tasks that have been accepted by a computing service provider. Another application can be found in the literature on job agents that focuses on sequencing problems. Mitra (2002), for instance, describes the following setting: There "is a large multi-unit firm with each unit in need of the facility provided by a particular repair and maintenance unit. The repair and maintenance unit can service only one unit at any given time. Therefore, units which remain unattended, incur a cost for the time they are down. In this framework, the firm's role is that of a planner wanting to service the units by forming a queue that minimises the total cost of waiting. Each unit's cost parameter is private information. The objective of the firm is to determine the order in which the units are to be serviced." Other applications, for example, arising in logistics or production processes like considered by Kovalyov and Pesch (2014) can rarely be found and might thus be an interesting field of study.

Finally, there are many additional potential research directions related to game theory and mechanism design aspects. There are, for example, only very few articles that consider risk averse agents. Moreover, as indicated in Sect. 2.1, it might be interesting to consider more general settings with regard to the categories of agents. For instance, one may let agents control multiple machines or jobs. Furthermore, the literature on job agents and uniform machine agents rarely considers multi-parameter settings. Finally, Tables 3, 4 and 5 reveal diverse literature gaps. One may, for example, analyze budget-balanced payments or the design of optimal mechanisms in case of machine agents. For specific problem settings, it is possible to consider different models of execution, constraints on the committed data, local objective functions, etc.

Acknowledgements This work has been supported by the German Science Foundation (DFG) through the Grant "Scheduling mechanisms for rail mounted gantries with regard to crane interdependencies" (PE 514/22-1).

References

Allahverdi A, Ng CT, Cheng TCE, Kovalyov MY (2008) A survey of scheduling problems with setup times or costs. Eur J Oper Res 187(3):985–1032

Ambrosio P, Auletta V (2005) Deterministic monotone algorithms for scheduling on related machines. In: Persiano G, Solis-Oba R (eds) Approximation and online algorithms, 2nd international workshop, WAOA 2004, revised selected papers, Springer, Berlin, pp 267–280

Ambrosio P, Auletta V (2008) Deterministic monotone algorithms for scheduling on related machines. Theor Comput Sci 406(3):173–186

Andelman N, Azar Y, Sorani M (2005) Truthful approximation mechanisms for scheduling selfish related machines. In: Diekert V, Durand B (eds) STACS 2005, 22nd annual symposium on theoretical aspects of computer science, Springer, Berlin, pp 69–82

Andelman N, Azar Y, Sorani M (2007) Truthful approximation mechanisms for scheduling selfish related machines. Theor Comput Syst 40(4):423–436

Angel E, Bampis E, Pascual F (2005) Truthful algorithms for scheduling selfish tasks on parallel machines. In: Deng X, Ye Y (eds) Internet and network economics, first international workshop, WINE 2005, proceedings, Springer, Berlin

Angel E, Bampis E, Pascual F (2006) Truthful algorithms for scheduling selfish tasks on parallel machines. Theor Comput Sci 369(1–3):157–168

Angel E, Bampis E, Pascual F, Tchetgnia AA (2009) On truthfulness and approximation for scheduling selfish tasks. J Sched 12(5):437–445

Angel E, Bampis E, Thibault N (2010) Randomized truthful algorithms for scheduling selfish tasks on parallel machines. In: López-Ortiz A (ed) LATIN 2010: theoretical informatics, 9th latin american symposium, proceedings, Springer, Berlin

Angel E, Bampis E, Thibault N (2012) Randomized truthful algorithms for scheduling selfish tasks on parallel machines. Theor Comput Sci 414(1):1–8

Archer A (2004) Mechanisms for discrete optimization with rational agents. PhD thesis, Cornell University

Archer A, Tardos É (2001) Truthful mechanisms for one-parameter agents. In: Proceedings of the 42nd IEEE symposium on foundations of computer science, IEEE, FOCS ’01, pp 482–491

Ashlagi I, Dobzinski S, Lavi R (2009) An optimal lower bound for anonymous scheduling mechanisms. In: Proceedings of the 10th ACM conference on electronic commerce, ACM, EC ’09, pp 169–176

Ashlagi I, Dobzinski S, Lavi R (2012) Optimal lower bounds for anonymous scheduling mechanisms. Math Oper Res 37(2):244–258

Auletta V, De Prisco R, Penna P, Persiano G (2004) Deterministic truthful approximation mechanisms for scheduling related machines. In: Diekert V, Habib M (eds) STACS 2004, 21st annual symposium on theoretical aspects of computer science, proceedings, Springer, Berlin, pp 608–619

Auletta V, Christodoulou G, Penna P (2012) Mechanisms for scheduling with single-bit private values. In: Serna M (ed) Algorithmic game theory, 5th international symposium, SAGT 2012, proceedings, Springer, Berlin, pp 25–36

Auletta V, Christodoulou G, Penna P (2015) Mechanisms for scheduling with single-bit private values. Theor Comput Syst 57(3):523–548

Azar Y, Hoefer M, Maor I, Reiffenhäuser R, Vöcking B (2015) Truthful mechanism design via correlated tree rounding. In: Proceedings of the 16th ACM conference on economics and computation, ACM, EC ’15, pp 415–432

Azar Y, Hoefer M, Maor I, Reiffenhäuser R, Vöcking B (2017) Truthful mechanism design via correlated tree rounding. Math Program 163(1–2):445–469

Błażewicz J, Ecker KH, Pesch E, Schmidt G, Węglarz J (2007) Handbook on scheduling: from theory to applications. Springer, Berlin

Boysen N, Fliedner M (2010) Cross dock scheduling: classification, literature review and research agenda. Omega 38(6):413–422

Boysen N, Fliedner M, Scholl A (2007) A classification of assembly line balancing problems. Eur J Oper Res 183(2):674–693

Boysen N, Fliedner M, Scholl A (2009) Sequencing mixed-model assembly lines: survey, classification and model critique. Eur J Oper Res 192(2):349–373

Brucker P, Drexl A, Möhring R, Neumann K, Pesch E (1999) Resource-constrained project scheduling: notation, classification, models, and methods. Eur J Oper Res 112(1):3–41

Chawla S, Hartline JD, Malec D, Sivan B (2013) Prior-independent mechanisms for scheduling. In: Proceedings of the 45th annual ACM symposium on theory of computing, ACM, STOC ’13, pp 51–60

Chen X, Du D, Zuluaga LF (2013) Copula-based randomized mechanisms for truthful scheduling on two unrelated machines. In: Vöcking B (ed) Algorithmic game theory, 6th international symposium, SAGT 2013, proceedings, Springer, Berlin, pp 231–242

Chen X, Du D, Zuluaga LF (2015) Copula-based randomized mechanisms for truthful scheduling on two unrelated machines. Theor Comput Syst 57(3):753–781

Christodoulou G, Koutsoupias E (2009) Mechanism design for scheduling. Bull EATCS 97:40–59

Christodoulou G, Kovács A (2010) A deterministic truthful PTAS for scheduling related machines. In: Proceedings of the 21st annual ACM-SIAM symposium on discrete algorithms, SIAM, SODA ’10, pp 1005–1016

Christodoulou G, Kovács A (2011) A global characterization of envy-free truthful scheduling of two tasks. In: Chen N, Elkind E, Koutsoupias E (eds) Internet and network economics, 7th international workshop, WINE 2011, proceedings, Springer, Berlin, pp 84–96

Christodoulou G, Kovács A (2013) A deterministic truthful PTAS for scheduling related machines. SIAM J Comput 42(4):1572–1595

Christodoulou G, Gourvès L, Pascual F (2007a) Scheduling selfish tasks: about the performance of truthful algorithms. In: Lin G (ed) Computing and combinatorics, 13th annual international conference, COCOON 2007, proceedings, Springer, Berlin, pp 187–197

Christodoulou G, Koutsoupias E, Kovács A (2007b) Mechanism design for fractional scheduling on unrelated machines. In: Arge L, Cachin C, Jurdziński T, Tarlecki A (eds) Automata, languages and programming, 34th international colloquium, ICALP 2007, proceedings, Springer, Berlin, pp 40–52

Christodoulou G, Koutsoupias E, Vidali A (2007c) A lower bound for scheduling mechanisms. In: Proceedings of the 18th annual ACM-SIAM symposium on discrete algorithms, ACM, SODA '07, pp 1163–1170

Christodoulou G, Koutsoupias E, Vidali A (2008) A characterization of 2-player mechanisms for scheduling. In: Halperin D, Mehlhorn K (eds) Algorithms—ESA 2008, 16th annual European symposium, proceedings, Springer, Berlin, pp 297–307

Christodoulou G, Koutsoupias E, Nanavati A (2009a) Coordination mechanisms. Theor Comput Sci 410(36):3327–3336

Christodoulou G, Koutsoupias E, Vidali A (2009b) A lower bound for scheduling mechanisms. Algorithmica 55(4):729–740

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

Clarke EH (1971) Multipart pricing of public goods. Public Choice 11(1):17–33

Daskalakis C, Weinberg SM (2015) Bayesian truthful mechanisms for job scheduling from bi-criterion approximation algorithms. In: Proceedings of the 26th annual ACM-SIAM symposium on discrete algorithms, ACM, SODA '15, pp 1934–1952

De P, Mitra M (2017) Incentives and justice for sequencing problems. Econ Theory 64(2):239–264

Dhangwatnotai P, Dobzinski S, Dughmi S, Roughgarden T (2008) Truthful approximation schemes for single-parameter agents. In: Proceedings of the 49th annual IEEE symposium on foundations of computer science, IEEE, FOCS '08, pp 15–24

Dhangwatnotai P, Dobzinski S, Dughmi S, Roughgarden T (2011) Truthful approximation schemes for single-parameter agents. SIAM J Comput 40(3):915–933

Dobzinski S, Dughmi S (2013) On the power of randomization in algorithmic mechanism design. SIAM J Comput 42(6):2287–2304

Dobzinski S, Sundararajan M (2008) On characterizations of truthful mechanisms for combinatorial auctions and scheduling. In: Proceedings of the 9th ACM conference on electronic commerce, ACM, EC '08, pp 38–47

Duives J, Heydenreich B, Mishra D, Müller R, Uetz M (2015) On optimal mechanism design for a sequencing problem. J Sched 18(1):45–59

Epstein L, van Stee R (2008) Maximizing the minimum load for selfish agents. In: Laber ES, Bornstein C, Nogueira LT, Faria L (eds) LATIN 2008: theoretical informatics, 8th latin american symposium, proceedings, Springer, Berlin, pp 264–275

Epstein L, van Stee R (2010) Maximizing the minimum load for selfish agents. Theor Comput Sci 411(1):44–57

Epstein L, Levin A, van Stee R (2013) A unified approach to truthful scheduling on related machines. In: Proceedings of the 24th annual ACM-SIAM symposium on discrete algorithms, ACM, SODA '13, pp 1243–1252

Epstein L, Levin A, van Stee R (2015) A unified approach to truthful scheduling on related machines. Math Oper Res 41(1):332–351

Ferrante A, Parlato G, Sorrentino F, Ventre C (2009) Fast payment schemes for truthful mechanisms with verification. Theor Comput Sci 410(8):886–899

Fudenberg D, Tirole J (1991) Game theory. MIT Press, Cambridge

Giannakopoulos Y, Kyropoulou M (2015) The VCG mechanism for Bayesian scheduling. In: Markakis E, Schäfer G (eds) Web and internet economics, 11th international conference, WINE 2015, proceedings, Springer, Berlin, pp 343–356

Graham RL, Lawler EL, Lenstra JK, Rinnooy Kan AHG (1979) Optimization and approximation in deterministic sequencing and scheduling: a survey. Ann Discrete Math 5:287–326

Groves T (1973) Incentives in teams. Econometrica 41(4):617–631

Hain R, Mitra M (2004) Simple sequencing problems with interdependent costs. Game Econ Behav 48(2):271–291

Hamers H, Klijn F, Suijs J (1999) On the balancedness of multiple machine sequencing games. Eur J Oper Res 119(3):678–691

Harks T, Klimm M, Möhring RH (2011) Characterizing the existence of potential functions in weighted congestion games. Theor Comput Syst 49(1):46–70

Hartline JD, Karlin AR (2007) Profit maximization in mechanism design. In: Nisan N, Roughgarden T, Tardos E, Vazirani VV (eds) Algorithmic game theory. Cambridge University Press, Cambridge, pp 331–361

Hashimoto K, Saitoh H (2012) Strategy-proof and anonymous rule in queueing problems: a relationship between equity and efficiency. Soc Choice Welf 38(3):473–480

Heydenreich B, Müller R, Uetz M (2007) Games and mechanism design in machine scheduling—an introduction. Prod Oper Manag 16(4):437–454

Heydenreich B, Mishra D, Müller R, Uetz M (2008) Optimal mechanisms for single machine scheduling. In: Papadimitriou C, Zhang S (eds) Internet and network economics, 4th international workshop, WINE 2008, proceedings, Springer, Berlin, pp 414–425

Hoeksma R, Uetz M (2013) Two dimensional optimal mechanism design for a sequencing problem. In: Goemans M, Correa J (eds) Integer programming and combinatorial optimization, 16th international conference, IPCO 2013, proceedings, Springer, Berlin, pp 242–253

Immorlica N, Li EL, Mirrokni VS, Schulz AS (2009) Coordination mechanisms for selfish scheduling. Theor Comput Sci 410(17):1589–1598

Kayı Ç, Ramaekers E (2010) Characterizations of Pareto-efficient, fair, and strategy-proof allocation rules in queueing problems. Game Econ Behav 68(1):220–232

Kayı Ç, Ramaekers E (2015) Corrigendum to characterizations of Pareto-efficient, fair, and strategy-proof allocation rules in queueing problems [Games Econ Behav 68(1) (2010) 220–232]. Game Econ Behav https://doi.org/10.1016/j.geb.2015.01.006

Koutsoupias E (2011) Scheduling without payments. In: Persiano G (ed) Algorithmic game theory, 4th international symposium, SAGT 2011, proceedings, Springer, Berlin, pp 143–153

Koutsoupias E (2014) Scheduling without payments. Theor Comput Syst 54(3):375–387

Koutsoupias E, Vidali A (2007) A lower bound of 1+ φ for truthful scheduling mechanisms. In: Kučera L, Kučera A (eds) Mathematical foundations of computer science 2007, 32nd international symposium, MFCS 2007, proceedings, Springer, Berlin, pp 454–464

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

Kovács A (2005) Fast monotone 3-approximation algorithm for scheduling related machines. In: Brodal GS, Leonardi S (eds) Algorithms—ESA 2005, 13th annual European symposium, proceedings, Springer, Berlin, pp 616–627

Kovács A (2006) Tighter approximation bounds for LPT scheduling in two special cases. In: Calamoneri T, Finocchi I, Italiano GF (eds) Algorithms and complexity, 6th Italian conference, CIAC 2006, proceedings, Springer, Berlin, pp 187–198

Kovács A (2009) Tighter approximation bounds for LPT scheduling in two special cases. J Discrete Algorithms 7(3):327–340

Kovalyov MY, Pesch E (2014) A game mechanism for single machine sequencing with zero risk. Omega 44:104–110

Kovalyov MY, Kress D, Meiswinkel S, Pesch E (2016) A parallel machine schedule updating game with compensations and zero risk. Working Paper, National Academy of Sciences of Belarus and University of Siegen

Kress D, Meiswinkel S, Pesch E (2017) Incentive compatible mechanisms for scheduling two-parameter job agents on parallel identical machines to minimize the weighted number of late jobs. Discrete Appl Math. https://doi.org/10.1016/j.dam.2017.08.026

Krishna V (2010) Auction theory, 2nd edn. Academic Press, Amsterdam

Lavi R (2007) Computationally efficient approximation mechanisms. In: Nisan N, Roughgarden T, Tardos E, Vazirani VV (eds) Algorithmic game theory. Cambridge University Press, Cambridge, pp 301–329

Lavi R, Swamy C (2007) Truthful mechanism design for multidimensional scheduling via cycle monotonicity. In: Proceedings of the 8th ACM conference on electronic commerce, ACM, EC '07, pp 252–261

Lavi R, Swamy C (2009) Truthful mechanism design for multidimensional scheduling via cycle monotonicity. Game Econ Behav 67(1):99–124

Leung JYT (ed) (2004) Handbook of scheduling: algorithms, models, and performance analysis. CRC Press, Boca Raton

Leung JYT, Li CL (2008) Scheduling with processing set restrictions: a survey. Int J Prod Econ 116(2):251–262

Lu P, Yu C (2008a) An improved randomized truthful mechanism for scheduling unrelated machines. In: Proceedings of the 25th international symposium on theoretical aspects of computer science, IBFI Schloss Dagstuhl, STACS '08, pp 527–538

Lu P, Yu C (2008b) Randomized truthful mechanisms for scheduling unrelated machines. In: Papadimitriou C, Zhang S (eds) Internet and network economics, 4th international workshop, WINE 2008, proceedings, Springer, pp 402–413

Mitra M (2001) Mechanism design in queueing problems. Econ Theory 17(2):277–305

Mitra M (2002) Achieving the first best in sequencing problems. Rev Econ Des 7(1):75–91

Mitra M (2005) Incomplete information and multiple machine queueing problems. Eur J Oper Res 165(1):251–266

Mu'alem A, Schapira M (2007) Setting lower bounds on truthfulness: extended abstract. In: Proceedings of the 18th annual ACM-SIAM symposium on discrete algorithms, ACM, SODA '07, pp 1143–1152

Mu'alem A, Schapira M (2017) Setting lower bounds on truthfulness. Working Paper

Nisan N (2007) Introduction to mechanism design (for computer scientists). In: Nisan N, Roughgarden T, Tardos E, Vazirani VV (eds) Algorithmic game theory. Cambridge University Press, Cambridge, pp 209–241

Nisan N, Ronen A (1999) Algorithmic mechanism design (extended abstract). In: Proceedings of the 31st annual ACM symposium on theory of computing, ACM, STOC '99, pp 129–140

Nisan N, Ronen A (2001) Algorithmic mechanism design. Game Econ Behav 35(1–2):166–196

Nisan N, Roughgarden T, Tardos E, Vazirani VV (2007) Algorithmic game theory. Cambridge University Press, Cambridge

Potts CN, Kovalyov MY (2000) Scheduling with batching: a review. Eur J Oper Res 120(2):228–249

Procaccia AD, Tennenholtz M (2009) Approximate mechanism design without money. In: Proceedings of the 10th ACM conference on electronic commerce, ACM, EC '09, pp 177–186

Roberts K (1979) The characterization of implementable choice rules. In: Laffont JJ (ed) Aggregation and revelation of preferences. North-Holland, Amsterdam, pp 321–348

Rosenthal RW (1973) A class of games possessing pure-strategy Nash equilibria. Int J Game Theory 2(1):65–67

Roughgarden T, Tardos E (2007) Introduction to the inefficiency of equilibria. In: Nisan N, Roughgarden T, Tardos E, Vazirani VV (eds) Algorithmic game theory. Cambridge University Press, Cambridge, pp 443–459

Schummer J, Vohra RV (2007) Mechanism design without money. In: Nisan N, Roughgarden T, Tardos E, Vazirani VV (eds) Algorithmic game theory. Cambridge University Press, Cambridge, pp 243–265

Suijs J (1996) On incentive compatibility and budget balancedness in public decision making. Econ Des 2(1):193–209

Vickrey W (1961) Counterspeculation, auctions, and competitive sealed tenders. J Finance 16(1):8–37

Vöcking B (2007) Selfish load balancing. In: Nisan N, Roughgarden T, Tardos E, Vazirani VV (eds) Algorithmic game theory. Cambridge University Press, Cambridge, pp 517–542