Serial Dictatorship Mechanism for Project Scheduling with Non-Renewable Resources

Péter Egri[*] and Tamás Kis[†]

Institute for Computer Science and Control, Hungarian Academy of Sciences,
Kende u. 13-17, 1111 Budapest, Hungary

Abstract

This paper considers a resource-constrained project scheduling problem with self-interested agents. A novel resource allocation model is presented and studied in a mechanism design setting without money. The novelties and specialties of our contribution include that the non-renewable resources are supplied at different dates, the jobs requiring the resources are related with precedence relations, and the utilities of the agents are based on the tardiness values of their jobs. We modify a classical scheduling algorithm for implementing the Serial Dictatorship Mechanism, which is then proven to be truthful and Pareto-optimal. Furthermore, the properties of the social welfare are studied.

Keywords Project scheduling, non-renewable resources, mechanism design without money, Serial Dictatorship Mechanism

1 Introduction

Recently, there has been a growing interest in game theoretical analysis in the supply chain management and scheduling research communities. In large-scale manufacturing systems with strategic setting, agents often possess private information, and since they are non-cooperative, they intend to manipulate the outcome of the system for their benefit. Allocation of multiple goods or resources is a frequently studied optimization problem of this sort. When the protocol controlling the system behavior includes monetary transfers, like in case of supply chains, setting the payments appropriately can be used to make manipulations ineffective [Egri and Váncza, 2012]. If such transfers are not allowed, usually only dictatorial mechanisms can prevent manipulations [see e.g. Abdulkadiroğlu and Sönmez, 1998]. In this paper, we study this latter situation specialized for a project scheduling application. The novelties and specialties of our contribution are: (i) the non-renewable (consumable) resources are supplied over the scheduling time horizon at different dates, (ii) the jobs requiring

[*]egri.peter@sztaki.mta.hu
[†]kis.tamas@sztaki.mta.hu

the resources are related with precedence relations, and (iii) the utilities of the agents are not arbitrary, but based on the tardiness values of their jobs.

More specifically, we consider a project scheduling problem with *non-renewable resource* constraints, where each project is owned by a selfish agent. The projects consist of *jobs* that compete for commonly used resources. In this paper we consider only non-renewable resources, such as raw-materials, energy, money [Gafarov et al., 2011, Grigoriev et al., 2005], but even computational resources—such as CPU, memory and network bandwidth—are frequently modeled as consumable resources in cloud infrastructures [see e.g. Kash et al., 2014]. The resources are consumed by the jobs and they have an initial stock which is replenished over time at given dates and in known quantities. The jobs have to be executed while meeting *precedence* and *resource constraints*. That is, each job may have some predecessors, all of which have to be completed prior to starting the job, and it may require some non-renewable resources which have to be on stock when starting the job, and once it is started, the stock levels of the respective resources are decreased by the required quantities. The stock levels can never be negative, so if the initial stock level of some resource is not enough to complete all the jobs, some of them may have to be delayed extra in order to meet the resource constraints. Each project has a *due date*, and if it is completed afterwards, it will be tardy. A *schedule* specifies the start time of each job, and it is *feasible* if all the precedence and resource constraints are satisfied (see Fig. 1 for an illustration). Throughout the paper we assume that the total supply from each resource equals the total demand in order to guarantee the existence of feasible schedules. Since the non-renewable resources are replenished over time, it is not obvious when to start the jobs when some optimization criteria are involved. In the basic problem (where all data is publicly known, and there are no selfish agents) a feasible schedule is sought in which the maximal tardiness among the projects is as small as possible. This problem can be efficiently solved by the method of Carlier and Rinnooy Kan [1982].

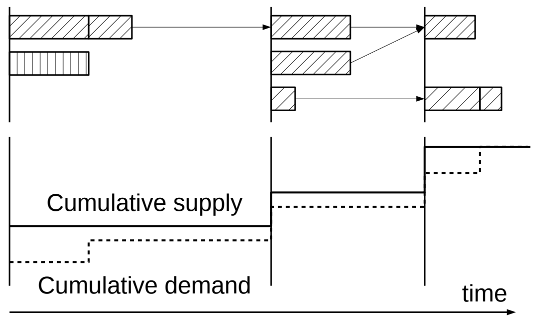


Figure 1: A sample schedule with 3 projects and a single resource.

In a multi-agent environment, projects are owned by selfish agents, which want to minimize the tardiness of their own projects. The due date of a project is only known by the corresponding agent. Further on, there is a *central in-*

ventory, which allocates the resources to the jobs of the projects over time. However, there is also a conflict of interests: while the central inventory still aims at minimizing the maximum tardiness over all projects (this makes the most unhappy agent less unhappy)[1], each agent is interested in minimizing its own tardiness. Therefore, the agents are competing for the resources, and they are inclined to be untruthful about their due dates in hope of achieving a more advantageous resource allocation for themselves. It is the central inventory, who can inspire the agents to tell their true due dates by using a *truthful allocation mechanism*, which ensures that reporting the true due dates yields the best outcome for each agent.

Main results of this paper. We investigate truthful mechanisms without payments for the above project scheduling problem. We will show that no truthful mechanism exists which always finds an optimal[2] solution. After this, we describe the *Serial Dictatorship Mechnaism* (SDM), which is (weakly) truthful, and always finds a Pareto-optimal solution. Our SDM is based on the polynomial time procedure of Carlier and Rinnooy Kan [1982] for solving the project scheduling problem (without agents). We will investigate the properties of the SDM, and among others, we will show that it is not able to find all Pareto-optimal solutions for the problem. We will also summarize computational results. Further on, we define a randomized SDM which can find any Pareto-optimal solution with positive probability.

The motivation for this research comes from real industrial production environments, where project leaders (the agents in the model) want to reserve the necessary resources greedily, in many cases too early, and in larger than necessary quantities, in order to finish their projects on time. Practical approaches like prioritizing the most important products or customers can help to alleviate the problem, but cannot guarantee any optimality criteria. This situation also resembles the coordination problem in supply chains, but a crucial difference is that in the latter appropriate payments can ensure truthfulness [see e.g. Egri and Váncza, 2013].

The paper is organized as follows. In Section 3, we review the classical scheduling model and its solution that will be the basis of the resource allocation problem. In Section 4, the mechanism design model is introduced, the impossibility of truthful and optimal mechanisms is proven, then the SDM for the this problem is described and analyzed. Next, we present a randomized version of the SDM in Section 5. Finally, in Section 6, we conclude the results and mention some future research directions.

[1]In the mechanism design literature this is referred to as *egalitarian social welfare*, which is considered more *fair* than minimizing the total tardiness, the *utilitarian social welfare* [see e.g. Rothe, 2015]. However, most of the results presented in this paper remains valid when this latter objective is considered instead.

[2]Throughout the paper we refer to optimality with respect to the objective of the central inventory.

2 Literature review

Planning assembly operations including precedence constraints and material supply is a relevant and frequently studied problem in production systems [Sternatz, 2015, Tiacci, 2015]. Most of these studies, like traditional optimization problems, usually assume a central decision maker and do not consider the conflict of interests. The most natural situation when multiple non-cooperative parties are involved occurs in supply chain material management problems [Edirisinghe and Atkins, 2017, Fandel and Trockel, 2011]. However, the game theoretical analysis and design is not limited to supply chains. Scheduling problems involving self-interested agents were already studied in the seminal work about algorithmic mechanism design [Nisan and Ronen, 2001], and even earlier [Váncza and Márkus, 2000]. Since then, several authors have combined scheduling and mechanism design [Christodoulou and Koutsoupias, 2009, Heydenreich et al., 2007], but most scheduling papers consider *renewable resources*—such as machines—as agents. A large number of mechanisms involve *payments* for incentivizing the agents in scheduling or allocation settings [Chen et al., 2016, Krysta et al., 2015, Robu et al., 2013]. Recently, mechanisms without money are also studied for scheduling problems [Giannakopoulos et al., 2016], but to the best of our knowledge, scheduling mechanisms with non-renewable resources and without payments are not yet investigated.

Our *resource allocation problem* is related to the one-sided matching problems without money, such as the *house allocation* and the *course allocation* [see e.g. Manlove, 2013]. These models consist of two different sets of objects, where the elements of one set (called *applicants*) have privately known preference orderings over the elements of the other set [Kurata et al., 2017]. This is in contrast with the two-sided matching problems, where the elements of both sets have preferences over the elements of the other set. The goal of the mechanism design for these problems is to give a matching between the two sets that satisfy certain properties, such as *truthfulness* and stability (e.g., *Pareto-optimality*)[3].

For these matching problems, a frequently used mechanism is the Serial Dictatorship Mechanism (SDM), which, in several cases, is the only mechanism satisfying the required properties, and furthermore, it is straightforward to implement [e.g. Abizada and Chen, 2016, Aziz and Mestre, 2014]. An SDM considers a—random or pre-existing—priority ordering of the applicants and works as follows. First, it determines the set of optimal allocations with regard to the preferences of the applicant with the highest priority. Then in each consecutive step, it takes the set from the previous step, and determines a subset of the best allocations considering the preferences of the next applicant. After the last applicant, it yields an allocation from the final set. Note that if the preferences always imply a unique preferred allocation, then only the preferences of the applicant with the highest priority (the dictator) matters, which is the classical dictatorship.

The house allocation problem is the one-to-one version of the one-sided

[3]These properties will be formally defined in Section 4.

matching, where each applicant can be paired with at most one house, and conversely, each house can be assigned to at most one applicant. For this problem the SDM is truthful, and in addition, it can generate every Pareto-optimal matchings—with different priority orderings—, and it is the only Pareto-optimal mechanism [Abdulkadiroğlu and Sönmez, 1998].

Dughmi and Ghosh [2010] study a one-sided, one-to-many General Assignment Problem (GAP) without money, and some of its special cases. In their model, job agents should be matched with capacitated machines. The problem is formulated as an integer program, and by relaxing the integrality constraints, an LP-based technique is shown to provide truthful approximate mechanisms.

The many-to-many extension of the one-sided matching is the course allocation problem, where both the applicants and the courses have quotas for their connections. The SDM can generate every Pareto-optimal matchings, however, it is truthful only in special cases [Cechlárová et al., 2016, Cechlárová and Fleiner, 2017]. Kash et al. present a dynamic version of the matching problem, where the agents are not present simultaneously, but can arrive any time, and their demands are not known in advance [Kash et al., 2014]. They regard renewable computational resources (such as CPU and memory), but they consider them *consumable*, i.e., once allocated, it is irrevocable, thus they are actually non-renewable.

Our resource allocation model is different from the above mentioned matching problems in several aspects. First of all, the preferences are not ordinal but cardinal, and not arbitrary: there is a scheduling problem in the background with a predefined structure that influences the preferences. For example, having a resource earlier is (weakly) preferred compared to having it later—if the goal is to minimize the tardiness. The matching also cannot be arbitrary, each job should be matched exactly with the required resources, only the timing can vary. Furthermore, contrary to the house and course allocation problems, the incoming batches of resources are divisible: they can be shared among several jobs. However, since satisfying only a part of the resource requirements has no value for the jobs, the problem resembles more to the matching than the *cake-cutting* models [see e.g. Brandt et al., 2016, Rothe, 2015].

Finally, we mention that if, in addition to non-renewable resources, the processing of jobs also require some renewable resources, such as machines, then quite a few results are known. Carlier [1984] was the first who studies machine scheduling problems with non-renewable resources, and further complexity results can be found in e.g., Grigoriev et al. [2005], Gafarov et al. [2011]. The approximability of machine scheduling problems is thoroughly studied for the makespan objective for single as well as parallel machine environments in Györgyi and Kis [2015a], Györgyi and Kis [2015b], Györgyi and Kis [2017], for the maximum lateness objective in Györgyi and Kis [2017], and for the total weighted completion time objective in Kis [2015].

3 The scheduling model

3.1 The project scheduling problem with non-renewable resources

Let us consider a set of projects P. Each project $p \in P$ has a due date d_p and a set of jobs J_p. Each job $j \in J_p$ has a processing time t_j. We assume that the J_p are disjoint and let J denote the union of all the J_p, containing altogether n jobs. Each project has a set of precedence relations $A_p \subset J_p \times J_p$, and if $(j,k) \in A_p$ then job j must be finished before k starts. We assume that the precedence relations induce an acyclic graph.

There is a set of non-renewable resources R, where each $\rho \in R$ has an initial supply of $b_{\rho,1}$ at time $u_1 = 0$, and additional supplies of $b_{\rho\ell}$ at times u_ℓ for $\ell = 2,\ldots,q$, where we assume that $u_1 < u_2 < \ldots < u_q$. Each job j requires a quantity of $a_{\rho j} \geq 0$ of resource $\rho \in R$ at its start.

We assume that for each resource $\rho \in R$ the demand does not exceed the supply, i.e., $\sum_{j\in J} a_{\rho j} \leq \sum_{\ell=1}^{q} b_{\rho\ell}$, otherwise the scheduling problem has no solution. For simplicity, we assume equality without loss of generality.

Let I denote an instance of the scheduling problem defined by the above introduced parameters.

We call $\mu = \{\mu_{\rho j\ell}\}$ an *allocation* of the supplied resources to the jobs, if for each resource ρ and time u_ℓ the supply $b_{\rho\ell}$ is divided among the jobs: $\sum_{j\in J} \mu_{\rho j\ell} = b_{\rho\ell}$. We call an allocation *feasible*, if every job j has enough resources allocated, i.e., $\forall \rho \in R : a_{\rho j} \leq \sum_{\ell=1}^{q} \mu_{\rho j\ell}$.[4] Henceforward we consider only feasible allocations and refer to them simply as allocations.

In order to have the schedule uniquely determined by an allocation, we assume that each job starts as early as possible, i.e., when (i) all the required resources are allocated to it, and (ii) every one of its predecessors defined by the precedence constraints are finished. Let us denote therefore the start time of job $j \in J_p$ w.r.t. allocation μ by

$$s_j^{(\mu)} = \min\left\{ s \geq 0 \;\middle|\; \forall \rho : \sum_{u_\ell \leq s} \mu_{\rho j\ell} \geq a_{\rho j} \wedge \forall (k,j) \in A_p : e_k^{(\mu)} \leq s \right\}, \tag{1}$$

where $e_k^{(\mu)}$ denotes the finish time of job k: $e_k^{(\mu)} = s_k^{(\mu)} + t_k$.

Finally, let $T_p^{(\mu)}$ denote the *tardiness* of project p as the non-negative difference between its due date and the maximal finish time of its jobs:

$$T_p^{(\mu)} = \max\left\{ \max_{j\in J_p} e_j^{(\mu)} - d_p, 0 \right\}. \tag{2}$$

3.2 The Carlier – Rinnooy Kan algorithm

Carlier and Rinnooy Kan [1982] gave a polynomial time algorithm for solving

[4]Since we assumed that the total supply equals the total demand of the resources, it is easy to see that equality holds in the definition of feasibility.

the above defined problem. We briefly recapitulate the main ideas of their solution here, since we are going to use a modified version of it in the SDM. Let us consider the graph defined by the jobs as nodes and precedence relations as edges, where the weight of edge (j,k) is t_j. Let $U(j)$ denote the set of all (direct or indirect) successors of job j, and W_{jk} the weight of the maximal path length between jobs j and $k \in U(j)$. For each project p, we define the cost function for each job $j \in J_p$ as $f_j(t) = \max\{\, t - d_p, 0\,\}$, i.e., the tardiness of the job j finishing at time t, with regard to the project's due date. In addition, let $B_\rho(u_\ell) = \sum_{\tau=1}^{\ell} b_{\rho\tau}$, the cumulative supply of resource ρ until time u_ℓ.

Then one can define a lower bound on the maximal tardiness in the case when job j starts at time u_ℓ:

$$\gamma_{\ell j} = \max\left\{ f_j(u_\ell + t_j), \max\{\, f_k(u_\ell + t_k + W_{jk}) \,|\, k \in U(j)\,\}\right\}. \tag{3}$$

The algorithm seeks the smallest γ (denoting the maximal tardiness), such that $\forall \rho, \ell : \sum_{j\in J}\{\, a_{\rho j} \,|\, \gamma < \gamma_{\ell j}\,\} \le B_\rho(u_{\ell-1})$, where $B_\rho(u_0) = 0$ (see Appendix A). For a fixed ℓ the smallest γ^*_ℓ can be found with a median search procedure, and the optimal $\gamma^* = \max_\ell \gamma^*_\ell$, for more details, see Carlier and Rinnooy Kan [1982].

Having the γ^*, the allocation μ can be computed by Algorithm 1.

Algorithm 1 Computing the allocation

```
Require: $\gamma^*$
  for $\ell = 2$ to $q$ do
    {Allocate resources to jobs that would be late starting at $u_\ell$}
    for $j : \gamma_{\ell j} > \gamma^* \wedge \gamma_{\ell-1 j} \le \gamma^*$ do
      Allocate the necessary resources to job $j$ arbitrarily from the resources
      arriving earlier than time $u_\ell$ and not yet allocated. (Due to the construc-
      tion of $\gamma^*$, there always exists enough free resources.)
    end for
  end for
  for $j : \gamma_{qj} \le \gamma^*$ do
    Allocate the necessary resources to job $j$ arbitrarily from the resources not
    yet allocated.
  end for
```

3.3 An example

In order to demonstrate the solution algorithm, let us consider a simple example with only one resource, two projects, four jobs and two supply times. Each job j $(1 \le j \le 4)$ has equal processing times $t_j = 1$ and requires one unit of the resource: $a_j = 1$, where we have omitted the index for the single resource. The precedences of the projects are as follows: $(j_1, j_2) \in A_{p_1}$ and $(j_3, j_4) \in A_{p_2}$, while their due dates are $d_p = 2$ $(1 \le p \le 2)$. There are two supply times, $u_1 = 0$

and $u_2 = 2$, both with two units of supplied materials: $b_\ell = 2$ $(1 \leq \ell \leq 2)$—again with omitted index for the resource. The resulted $\gamma_{\ell j}$ values are shown in Table 1.

		j			
		1	2	3	4
ℓ	1	0	0	0	0
	2	2	1	2	1

Table 1: The $\gamma_{\ell j}$ values for the example

Then the algorithm can compute $\gamma_1^* = 0$ and $\gamma_2^* = 1$, from which $\gamma^* = 1$, i.e., the optimal schedule will result in one time unit tardiness. This schedule is when jobs j_1 and j_3 receives their required resources at time u_1, and the rest at time u_2.

4 Mechanism design for project scheduling

In the mechanism design problem we consider project agents with their due dates as private information. All other information is assumed to be public knowledge.[5] We examine the problem of a central inventory, which has to allocate the resources supplied over time to the jobs.

We seek a *direct revelation* mechanism that consists of two steps: (i) collecting due date information from the projects, and (ii) allocating resources to the jobs. Since the project agents are interested in their own tardiness, they might report false due dates to the central inventory in order to influence the allocation to their advantage. We refer to the reported due dates as d'_p. For practical reasons, we restrict our study to mechanisms without money, i.e., it is not allowed to charge higher prices for resources arriving earlier.

Definition 1 (Mechanism)**.** *Let I denote a scheduling problem instance. A deterministic resource allocation mechanism is a function mapping the problem instance to an allocation: $\Phi(I) = \mu$.*

Definition 2 (Preference)**.** *Project p prefers allocation μ to μ' ($\mu \succ_p \mu'$), if $T_p^{(\mu)} < T_p^{(\mu')}$, and weakly prefers μ to μ' ($\mu \succeq_p \mu'$), if $T_p^{(\mu)} \leq T_p^{(\mu')}$.*

An important property of a mechanism is *truthfulness*, when the agents cannot decrease their resulted tardiness by misreporting the due dates. Truthfulness is a desired property of a mechanism, since with false (usually too early) due dates the inventory has no hope to take the tardinesses into consideration.

[5]This restriction is not necessary, only assumed for keeping the model simple. The set of private information can be extended to every parameter related to the projects. In this case, one does not have to use a *direct* mechanism—i.e., where the agents should report the full private information—only the $a_{\rho j}$ and $\gamma_{\ell j}$ values are required by the mechanism.

Definition 3 (Truthfulness). *Let I denote an arbitrary scheduling problem instance and I'_p the same problem, but with due date d'_p of project p instead of d_p. A mechanism Φ is* truthful*, if for each instance I, project p, and due date d'_p: $\Phi(I) \succeq_p \Phi(I'_p)$.*

Note that the definition uses weak preference, thus reporting a false due date does not necessarily worsen the tardiness of a project[6]. However, we assume *benevolent* agents henceforward, i.e., they report truthfully, if they cannot decrease their tardiness by misreporting.

Since in mechanism design it is often impossible to guarantee an optimal solution, this is the situation in our model as we will see in Subsection 4.1, therefore frequently weaker criteria are considered instead. A widely used property for characterizing an acceptable solution is the *Pareto-optimality*, when the resulted allocation cannot be improved for any agent without damaging the others.

Definition 4 (Pareto-optimality). *An allocation μ* Pareto-dominates *μ', if $\forall p : \mu \succeq_p \mu'$ and $\exists p : \mu \succ_p \mu'$. An allocation μ is* Pareto-optimal*, if no other allocation Pareto-dominates it. A mechanism is Pareto-optimal, if for all inputs it yields a Pareto-optimal allocation.*

Note that not every optimal allocation is Pareto-optimal, if we consider the maximal tardiness, not the sum of the tardinesses. However, if an allocation μ *Pareto-dominates* μ', then the maximal tardiness implied by μ cannot be greater than that of μ'. This property guarantees that there is at least one optimal allocation among the Pareto-optimal ones.

4.1 Impossibility of truthful and optimal mechanisms

The first fundamental question we address is whether our scheduling problem admits a mechanism which ensures that the benevolent players always tell their true due dates, and always finds an optimal solution for the scheduling problem (one minimizing the maximal tardiness of the projects)?

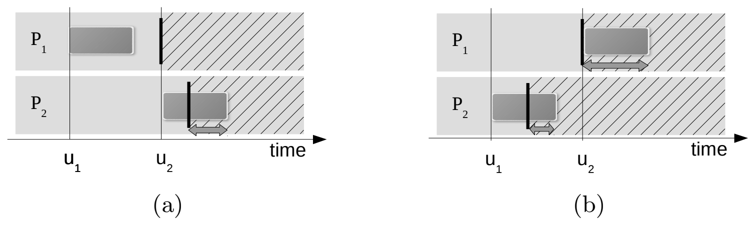


Figure 2: The optimal schedules in two different problem instances.

[6]Requiring strict preference would be problematic for the existence of truthful mechanisms. For example, if a project has an appropriately late due date, any feasible allocation results in no tardiness for the project, thus reporting any due date results in the same zero tardiness for the agent.

Proposition 1. *There is no truthful mechanism that always returns an optimal allocation.*

Proof. By contradiction, suppose we have a mechanism that is truthful, and on all inputs returns an optimal solution to the scheduling problem. Now we examine how it works on the following problem instance I. There are only two projects, p_1 and p_2, consisting of one job each, j_1 and j_2, respectively, and a single resource ρ with an initial supply of $b_{\rho 1} = 1$ at $u_1 = 0$, and a second supply of $b_{\rho 2} = 1$ at $u_2 = 4$. The two jobs are identical, i.e., $t_{j_1} = t_{j_2} = 3$, and $a_{\rho j_1} = a_{\rho j_2} = 1$, but project p_1 has a due-date of $d_{p_1} = 4$, and project p_2 has a due-date of $d_{p_2} = 5$. Notice that in any feasible schedule at most one job may start at $u_1 = 0$, the other must wait for the second supply at u_2. Suppose both projects report their true due dates. Then the algorithm must find the unique optimum in which j_1 starts at u_1, and j_2 starts at u_2. The tardiness of p_1 is then 0, and that of p_2 is 2 time units. This is depicted in Figure 2a.

Now suppose p_2 is not truthful, and reports $d'_{p_2} = 2$ instead of its original due date. On the one hand, since the algorithm is truthful, it should not modify the optimal solution obtained for the true values (as there is a unique optimum for the true due-dates). On the other hand, consider the problem instance I' which differs from I only in the due date of p_2, which is 2 in I'. Then, the algorithm must return the unique optimum for this instance, in which job j_1 starts at u_2, and job j_2 starts at u_1, giving a tardiness of 3 for project p_1, and 1 for project p_2, see Figure 2b.

Since the algorithm has no information about the private due dates of the projects, it cannot tell, whether p_2 lies about its due date, or tells the truth. So, it cannot be truthful and optimal at the same time, a contradiction. □

Note that the claim of the proposition remains valid if we change the optimality criterion to the total tardiness instead of the maximal tardiness.

Notice that the above impossibility result is not concerned with the complexity of the optimization algorithm, which makes it even more general.

Corollary 1. *The naïve mechanism in which the central inventory computes an optimal allocation with the Carlier–Rinnooy Kan algorithm is not truthful.*

This corollary implies that if the central inventory uses the naïve mechanism, it is beneficial for the projects to report as early due date as possible. This corresponds to the industrial practice when everyone requests the resources as soon as possible.

4.2 Serial Dictatorship Mechanism

Instead of minimizing the maximal tardiness as the naïve mechanism, the SDM considers a multi-objective optimization problem. It requires a priority ordering of the projects and generates an allocation that results in the lexicographically smallest vector of the job tardiness values. For the sake of simplicity we assume that the priority ordering is a fixed, commonly known input of the mechanism.

The basic idea of the mechanism is to take the projects in decreasing order of priority. In step 1, it takes the project p_1 with the highest priority, and creates an allocation $\mu^{(1)}$ that minimizes ${T'}_{p_1}^{(\mu^{(1)})}$, where T' denotes the tardiness function of Eq. 2 considering the reported due dates instead of the real ones. Then in each subsequent step k, a new allocation $\mu^{(k)}$ is computed that minimizes ${T'}_{p_k}^{(\mu^{(k)})}$, with the constraints that it cannot increase the tardinesses of the projects with higher priorities, i.e., $\forall k' \in \{1, \dots, k-1\} : {T'}_{p_{k'}}^{(\mu^{(k)})} \leq {T'}_{p_{k'}}^{(\mu^{(k-1)})}$. The resulted tardinesses of projects with lower priorities than p_k are completely disregarded in step k.

The allocation $\mu^{(k)}$ can be computed with a modified version of the Carlier–Rinnooy Kan algorithm. In step k, instead of $\gamma_{\ell j}$, we use the following $\gamma_{\ell j}^{(k)}$:

$$\gamma_{\ell j}^{(k)} = \begin{cases} \gamma'_{\ell j}, & j \in J_{p_k} \\ \infty, & j \in J_{p_{k'}} \wedge k' < k \wedge \gamma'_{\ell j} > {T'}_{p_{k'}}^{(\mu^{(k-1)})} \\ 0 & \text{otherwise} \end{cases}, \tag{4}$$

where $\gamma'_{\ell j}$ is defined by Eq. 3, but considering the reported d'_p due dates in the cost function f instead of the real ones. For the jobs of project p_k, this involves the lower bounds $\gamma'_{\ell j}$ for the tardiness, while for any other job it is either zero or infinity. For projects considered before p_k, any allocation that would result in larger tardiness for them than in the previous step, infinite tardiness is used, these are therefore cannot start at u_ℓ or later. For the remaining case (when $\gamma_{\ell j}^{(k)}$ is defined as 0), the jobs may start at u_ℓ without increasing the tardiness of the corresponding project.

Similarly to the original algorithm, we are looking for the smallest $\gamma^{(k)}$, such that $\forall \rho, \ell : \sum_{j \in J} \{ a_{\rho j} \mid \gamma^{(k)} < \gamma_{\ell j}^{(k)} \} \leq B_\rho(u_{\ell-1})$, see Algorithm 2.

Algorithm 2 Serial Dictatorship Mechanism

```
Require: $p_1, \dots, p_n$: an arbitrary priority ordering of the projects
  The projects announce their due dates to the central inventory
  for $k = 1$ to $n$ do
    for $\ell = 1$ to q do
      Compute the $\gamma_{\ell j}^{(k)}$ values
      $\gamma_\ell^{(k)*} := \min \left\{ \gamma^{(k)} \mid \forall \rho : \sum_{j \in J} \{ a_{\rho j} \mid \gamma^{(k)} < \gamma_{\ell j}^{(k)} \} \leq B_\rho(u_{\ell-1}) \right\}$
    end for
    $\gamma^{(k)*} := \max_\ell \gamma_\ell^{(k)*}$
    Compute $\mu^{(k)}$[7]
  end for
  Allocate the resources according to $\mu^{(n)}$.
```

[7]This is not necessary, since only the tardinesses of projects $p_1, \dots, p_k$ are used in the next step. For project p_k this will be equal to $\gamma^{(k)*}$, while for the other projects the tardinesses remain the same as in the previous step.

Theorem 1. *The SDM is truthful.*

Proof. Let us consider an arbitrary project p_k. In steps $1, \dots, k-1$, the due date d'_{p_k} is disregarded by the mechanism, therefore reporting it falsely cannot decrease the tardiness of p_k. In step k, the mechanism minimizes the tardiness of p_k with respect to the constraints derived from the previous steps, and using the reported due date of p_k. We claim that reporting a false due date cannot decrease the tardiness of p_k. For suppose, p_k reports a false due date $d'_{p_k} < d_{p_k}$ and let ${\mu'}^{(k)}$ and $\mu^{(k)}$ denote the corresponding allocations. If the tardiness of p_k is smaller with respect to ${\mu'}^{(k)}$ than that for $\mu^{(k)}$, then ${\mu'}^{(k)}$ would be a better allocation for p_k even when reporting its true due date, which is a contradiction. In steps $k' = k+1, \dots, n$, the tardiness ${T'}_{p_k}^{(\mu^{(k')})} = {T'}_{p_k}^{(\mu^{(k)})}$ remains constant: it cannot increase due to the construction of the mechanism, but it also cannot decrease, otherwise $\mu^{(k)}$ is not optimal in step k, which is a contradiction. $\square$

Note that the proof of truthfulness requires that the agents cannot influence the priority ordering. From now on, we take advantage of its truthfulness, and assume that the SDM possesses the real due dates. Let us prove the Pareto-optimality of the mechanism.

Theorem 2. *The SDM is Pareto-optimal.*

Proof. Let's indirectly assume that $\exists \mu'$ Pareto-dominating $\mu^{(n)}$, i.e., $\forall p \in P : \mu' \succeq_p \mu^{(n)}$ and $\exists p_k : \mu' \succ_{p_k} \mu^{(n)}$. This contradicts optimality of $\mu^{(k)}$ in step k, thus such μ' cannot exist. $\square$

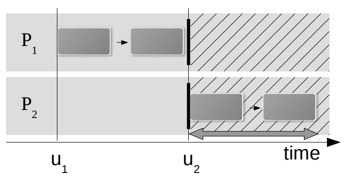
Corollary 2. *If there is a schedule where no project is tardy, then the SDM returns such a schedule.*

For several matching problems, every Pareto-optimal solutions can be generated by an SDM by using different priority orderings. Unfortunately, this does not hold for our resource allocation problem. As a consequence, although there exists at least one optimal allocation among the Pareto-optimal ones, it is possible that such allocations cannot be found by any SDM.

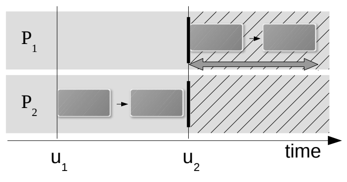
Proposition 2. *Not every Pareto-optimal solution can be generated by an SDM.*

Proof. Let us consider a simple scheduling problem with two projects of two jobs each, one resource and two supply times. Let $d_{p_1} = d_{p_2} = u_2$, $a_{\rho j_1} = a_{\rho j_2} = a_{\rho j_3} = a_{\rho j_4} = 1$, $A_{p_1} = \{ (j_1, j_2) \}$, $A_{p_2} = \{ (j_3, j_4) \}$, $b_{\rho 1} = b_{\rho 2} = 2$, and $t_{j_1} = t_{j_2} = t_{j_3} = t_{j_4} = (u_2 - u_1)/2$.

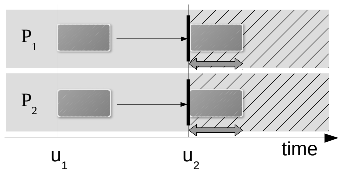
There are only 2 SDMs for this problem, but 3 Pareto-optimal solutions shown in Fig. 3. The two possible orderings of the projects for the SDM result in (maximal) tardiness $t_{j_1} + t_{j_2} = t_{j_3} + t_{j_4}$, illustrated in Figs 3a and 3b. However, the allocation shown on Fig. 3c is also Pareto-optimal and its maximal tardiness is the half of what is achievable with an SDM. $\square$



(a) p_1 with the higher priority



(b) p_2 with the higher priority



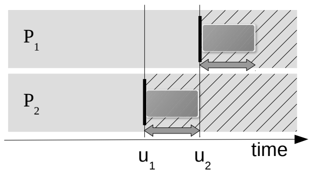
(c) nondictatorial optimal solution

Figure 3: Pareto-optimal solutions of the problem.

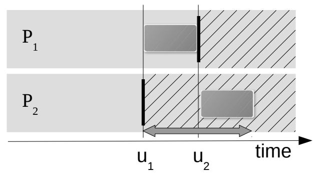
Proposition 2 implies that using SDMs may exclude the possibility of generating an optimal solution—despite always being Pareto-optimal. Unfortunately, there is an even more serious drawback of the SDMs. As the next theorem shows, the difference between the optimal and the maximal tardiness generated by an SDM is unbounded.

Proposition 3. *The maximal tardiness found by the SDM can be arbitrary far from the optimal.*

Proof. Let us consider a simple scheduling problem with two projects of one job each, one resource and two supply times. Let $d_{p_1} = u_2$, $d_{p_2} = u_1$, $a_{\rho j_1} = a_{\rho j_2} = 1$, $b_{\rho 1} = b_{\rho 2} = 1$ and $t_{j_1} = t_{j_2}$ (a fixed constant).



(a) The optimal solution



(b) Solution of the (p_1, p_2)-SDM.

Figure 4: Two possible allocations for the problem.

Fig. 4a illustrates the optimal schedule for this case, when the job of the second project gets the resource at u_1 and the other job at u_2. This result in tardinesses for both projects equal to their processing times. The solution on Fig. 4b is resulted by an SDM where p_1 has the higher priority. In order to avoid (or minimize) its tardiness, the job of p_1 must get the resource arriving at u_1. This results in $u_2 - u_1 + t_{j_2}$ maximal tardiness at project p_2.

As $u_2 \to \infty$, the maximal tardiness resulted by the optimal allocation does not change, but with the SDM it grows infinitely. $\square$

Note that the claim of Proposition 3 remains valid if we change the optimality criterion to the total tardiness instead of the maximal tardiness.

In order to compare the optimum of the scheduling problem with the one obtained by SDM, we shift the tardiness values of the schedules, which is a common technique in scheduling theory **??**. That is, the *shifted tardiness value* of a schedule s is

$$T_s^{\Delta} := T_s + u_q.$$

The shifted tardiness of any feasible schedule is u_q or more. Let $T_{\text{opt}}^{\Delta} := T_{\text{opt}} + u_q$ denote the optimum tardiness increased by u_q. The *relative error* of some schedule s is

$$\mathsf{Rel}(s) := \frac{T_s^{\Delta}}{T_{\text{opt}}^{\Delta}}. \tag{5}$$

By this formula, the relative error of an optimal schedule is 1. The following easy observation shows that with this normalized objective function, the relative error of those schedules obtained by SDM is at most 2.

Proposition 4. *The relative error of any schedule computed by SDM is at most 2.*

Proof. In order to prove the statement, we define a trivial feasible schedule with a relative error of at most 2, and argue that no job in a schedule obtained by SDM starts later than the same job in the trivial schedule.

In the trivial schedule s_{trivial} all the jobs of all the projects are started at time u_q or later if they have some predecessors. More precisely, in the trivial schedule first we schedule all the jobs without any predecessors at time u_q, then we schedule their immediate successors at the earliest possible time without violating the precedence constraints, etc. (or in other words, we schedule the jobs in topological order from time u_q on without any unnecessary delays). The trivial schedule satisfies all the precedence constraints by construction, and all the resource constraints as well, since by time u_q, all the resources are supplied, and the total supply equals the total demand for each resource by assumption.

Now consider the output of SDM. Since it is Pareto-optimal, no job may be started earlier without violating a resource constraint, or a precedence constraint. Hence, in this schedule no job may be started later than the same job in the trivial schedule.

Finally, we claim that the relative error of the trivial schedule is at most 2. On the one hand, in any feasible schedule, the tardiness of any project is at least u_q as we have already noted. Now consider the optimal schedule s_{opt}, and increase the start time of each job by u_q. In the the resulting schedule s', every job starts after u_q. Notice that in s', no job starts before the same job in the trivial schedule. Since the tardiness of each project is increased by u_q in s', we conclude that

$$\mathsf{Rel}(s_{\text{trivial}}) = \frac{T^{\Delta}_{\text{trivial}}}{T^{\Delta}_{\text{opt}}} \le \frac{T^{\Delta}_{s'}}{T^{\Delta}_{\text{opt}}} = \frac{T^{\Delta}_{\text{opt}} + u_q}{T^{\Delta}_{\text{opt}}} \le \frac{T_{\text{opt}} + 2u_q}{T_{\text{opt}} + u_q} \le 2.$$

□

Note that the tardiness value is shifted in order to avoid zero value in the denominator of the relative error. Another possibility to do this is to compare the resulted error to the optimal tardiness with the following formula:

$$\frac{T_{\text{sdm}} - T_{\text{opt}}}{\max\{T_{\text{opt}}, 1\}}. \tag{6}$$

Due to Corollary 2, if $T_{\text{opt}} = 0$ then this equals zero, otherwise—assuming integer parameters—it reduces to $(T_{\text{sdm}} - T_{\text{opt}})/T_{\text{opt}}$. However, we will consider the relative error defined by Eq. 5 hereafter.

We can also get an upper bound on the absolute error of the schedule resulted by the SDM. In order to do this, let us define a relaxed problem without resource constraints. The optimal solution for this problem, s_{relaxed}, is when each job starts as early as possible: those jobs that do not have predecessors start at $u_1 = 0$, while others start as soon as their predecessors are finished. Note that in s_{relaxed} every job starts exactly u_q time unit earlier than in s_{trivial}. Thus, if T_{relaxed} and T_{sdm} denote the maximal tardiness of s_{relaxed} and a schedule resulted by an SDM, respectively, we have $T_{\text{relaxed}} \le T_{\text{opt}} \le T_{\text{sdm}} \le T_{\text{trivial}} \le T_{\text{relaxed}} + u_q$. By rearranging these inequalities, we get an upper bound for the absolute error:

$$T_{\text{sdm}} - T_{\text{opt}} \le u_q. \tag{7}$$

This can be used to measure the relation of the absolute error and its upper bound by $(T_{\text{sdm}} - T_{\text{opt}})/u_q$, which yields a value between 0 and 1.

4.3 Numerical study

In order to asses the performance of SDM in practice, we have conducted a series of computational experiments. To this end, we have generated several problem instances with various characteristics, and compared the maximal tardinesses obtained by the Carlier – Rinnooy Kan algorithm and by the SDM with a random priority ordering. For comparison, we used the relative error defined by formula (5).

We have generated problem instances with $|P| \in \{10, 50, 100\}$ projects and $q \in \{5, 10, 15\}$ supply dates. In all instances the number of jobs in each project was $|J_p| = 5$. The project parameters were random numbers, i.e., $d_p \sim U(1, 50)$ for each project p, $t_j \sim U(1, 5)$ and $a_{\rho j} \sim U(0, 5)$ for all the jobs j and resources ρ, where $U(a, b)$ denotes the discrete uniform distribution on the interval $[a, b]$. The density of the precedence graph of each project was 0.2. The supplies were generated with $u_1 = 1$, $(u_\ell - u_{\ell-1}) \sim U(1, 50/q)$, $b_{\rho\ell} \sim U(0, \sum_j a_{\rho j} - B_\rho(u_{\ell-1}))$, and $b_{\rho q} = \sum_j a_{\rho j} - B_\rho(u_{q-1})$. The results show how the relative error varies depending on $|P|$, $|R|$ and q. Each value in Table 2 represents an average (or maximum) over 1000 problem instances.

		Resources ($\lvert R\rvert$)					
		1	2	4	6	8	10
q	5	103%	106%	108%	110%	110%	112%
	10	101%	101%	102%	103%	102%	103%
	15	100%	101%	101%	101%	101%	101%

(a) Average with $|P| = 10$ projects

		Resources ($\lvert R\rvert$)					
		1	2	4	6	8	10
q	5	181%	181%	181%	174%	182%	171%
	10	168%	167%	151%	151%	152%	161%
	15	154%	149%	134%	155%	129%	156%

(b) Maximum with $|P| = 10$ projects

		Resources ($\lvert R\rvert$)					
		1	2	4	6	8	10
q	5	104%	107%	111%	114%	117%	118%
	10	101%	102%	103%	104%	104%	104%
	15	100%	101%	101%	101%	102%	102%

(c) Average with $|P| = 50$ projects

		Resources ($\lvert R\rvert$)					
		1	2	4	6	8	10
q	5	172%	173%	175%	183%	176%	171%
	10	131%	132%	149%	135%	149%	150%
	15	116%	120%	125%	118%	117%	123%

(d) Maximum with $|P| = 50$ projects

		Resources ($\lvert R\rvert$)					
		1	2	4	6	8	10
q	5	105%	108%	112%	115%	119%	119%
	10	101%	102%	103%	104%	104%	106%
	15	100%	101%	101%	101%	102%	102%

(e) Average with $|P| = 100$ projects

		Resources ($\lvert R\rvert$)					
		1	2	4	6	8	10
q	5	166%	176%	168%	170%	175%	171%
	10	136%	168%	135%	139%	138%	140%
	15	125%	120%	123%	115%	120%	126%

(f) Maximum with $|P| = 100$ projects

Table 2: Average and maximum relative errors

Tables 2a, 2c and 2e suggest that there are two ways to decrease the expected error: with more frequent supplies or with less resources. When the number of supplies increases, there are usually more opportunities to schedule the non-tardy projects closer to their due dates, thus freeing some resources for the low priority projects. Decreasing the number of resources seems to be difficult in practice, but only the scarce resources are relevant for the problem. If the inventory keeps enough *safety stock*, then that resource does not constrain the schedule, thus it can be omitted from the model. Of course, both approaches come at a price which should be considered and balanced with the estimated cost of the tardiness.

Tables 2b, 2d and 2f presents the maximum error considering the same instances as for the average. Similarly to the average case, the maximum error also decreases when q increases. Furthermore, it can be observed that the maximum error tends to decrease with more projects. Since these values are the extreme cases, it is more difficult finding trends in these tables, but they can be used for estimating the worst case scenarios.

5 SDM with random endowments

In order to remedy the negative consequences of Proposition 2, we introduce a randomized extension of the SDM in this section.

Definition 5 (Randomized mechanism [Nisan and Ronen, 2001])**.** *A randomized mechanism is a probability distribution over a family* $\{ \Phi_r \}$ *of deterministic* *mechanisms. A randomized mechanism is called* truthful *(*Pareto-optimal*), if each deterministic mechanism in its support is truthful (Pareto-optimal).*

Let us modify the SDM such that it starts with a random (feasible) allocation $\mu^{(0)}$, and in each step k it makes a Pareto-improvement on it, resulting in allocation $\mu^{(k)}$. This randomized mechanism can be interpreted as follows: given an r random allocation, the Φ_r is a deterministic mechanism that executes Pareto-improvements on the allocation r according to the given priority ordering. Then the SDM with random endowments (SDMRE) is a probability distribution over $\{ \Phi_r \}$.

Algorithm 3 Computing a random allocation

```
for $\ell = 1$ to $q$ do
  for $\rho \in R$ do
    while $b_{\rho\ell} > 0$ do
      Let $j$ be a random job such that $a_{\rho j} > 0$
      Let $\mu_{\rho j \ell} = \min\{ a_{\rho j}, b_{\rho\ell} \}$
      Decrease $a_{\rho j}$ and $b_{\rho\ell}$ with the allocated quantity $\min\{ a_{\rho j}, b_{\rho\ell} \}$
    end while
  end for
end for
```

The initial allocation can be computed for example with Algorithm 3. Note that this algorithm cannot result in all possible feasible allocations: whenever a supply and a demand is chosen, either the whole demand will be covered with the allocation or the whole supply will be allocated for that demand. It is easy to see however, that any allocation can be transformed into an allocation that can be the output of Algorithm 3 and they both result in the same schedule (start times). Therefore this method does not exclude any significant solutions.

The allocation $\mu^{(k)}$ can be computed with a modified version of the SDM algorithm, where in step k we use the following $\gamma_{\ell j}^{(k)}$:

$$\gamma_{\ell j}^{(k)} = \begin{cases} \gamma'_{\ell j}, & j \in J_{p_k} \\ \infty, & j \in J_{p_{k'}} \wedge k' \neq k \wedge \gamma'_{\ell j} > {T'}_{p_{k'}}^{(\mu^{(k-1)})} \\ 0 & \text{otherwise} \end{cases} . \tag{8}$$

For project p_k, this takes the lower bounds of the tardiness, while for any other job it is either zero or infinity. For projects other than p_k, any allocation that would result in larger tardiness for them than in the previous step, infinite tardiness is considered, these are therefore excluded from an optimal solution. Any other allocation is allowed, thus they cause no tardiness in this step. This means that the algorithm minimizes the tardiness of project p_k, while enforces an upper bound on the other projects' tardinesses.

Similarly to the SDM, we are looking for the smallest $\gamma^{(k)}$, such that $\forall \rho, \ell :$ $\sum_{j \in J} \{ a_{\rho j} \mid \gamma^{(k)} < \gamma_{\ell j}^{(k)} \} \leq B_\rho(u_{\ell-1})$, see Algorithm 4.

Algorithm 4 SDM with Random Endowments (SDMRE)

```
Require: $p_1, \ldots, p_n$: an arbitrary ordering of the projects
  Let $\mu^{(0)}$ be a random (feasible) allocation
  The projects announce their due dates to the central inventory
  for $k = 1$ to $n$ do
    for $\ell = 1$ to q do
      Compute the $\gamma_{\ell j}^{(k)}$ values
      $\gamma_{\ell}^{(k)*} := \min \left\{ \gamma^{(k)} \mid \forall \rho : \sum_{j \in J} \{ a_{\rho j} \mid \gamma^{(k)} < \gamma_{\ell j}^{(k)} \} \leq B_\rho(u_{\ell-1}) \right\}$
    end for
    $\gamma^{(k)*} := \max_\ell \gamma_\ell^{(k)*}$
    Compute $\mu^{(k)}$
  end for
  Allocate the resources according to $\mu^{(n)}$
```

Theorem 3. *The SDMRE is truthful.*

Proof. Let us consider an arbitrary step k of the mechanism. Since the algorithm minimizes the tardiness of p_k in this step, it cannot benefit from a false d'_{p_k}. For any other $k' \neq k$, the $T'^{(\mu^{(k-1)})}_{p_{k'}}$ tardiness serves only as a constraint on the $T'^{(\mu^{(k)})}_{p_{k'}}$, which both change similarly depending on $d_{p'_k}$. Therefore also $p_{k'}$ cannot benefit from reporting a false due date. □

Note that the proof of truthfulness requires that the agents can influence neither the priority ordering nor the initial allocation. It is easy to see that the SDMRE is also Pareto-optimal, and in addition, since the initial allocation is arbitrary (any Pareto-optimal allocation can be generated with positive probability), the following theorem is true:

Theorem 4. *An allocation is Pareto-optimal if and only if it can be resulted by an SDMRE.*

Note however, that since every Pareto-optimal solution can be the output of the SDMRE, the claim of Proposition 3 is still valid for this mechanism. Furthermore, the numerical studies have shown that the resulted relative errors of the SDMRE are similar to those of the SDM presented in Table 2, therefore the average performance of the two approaches are not significantly different.

6 Conclusions

In this paper we introduced the non-renewable resource allocation problem for project scheduling and examined the properties of the SDM and SDMRE in this setting. We proved their truthfulness and Pareto-optimality, and identified some of their limitations considering optimality.

It would be interesting to investigate realistic special cases for the scheduling problem. For example, the supply of resources is usually not random, but follows some pattern resulted from the applied ordering policy, such as the *fixed order quantity* or *fixed time period*. Another possibility is to consider similar projects, which occurs when the products with different *features* define almost identical projects with slightly different resource requirements. The model also could be extended, e.g., with renewable resource constraints, for which case the computational complexity introduces additional challenges.

Acknowledgments

This work has been supported by the Hungarian Scientific Research Fund (OTKA) Grant No. 112881 and by the GINOP-2.3.2-15-2016-00002 grant on an “Industry 4.0 Research and Innovation Center of Excellence”.

References

References

A. Abdulkadiroğlu and T. Sönmez. Random serial dictatorship and the core from random endowments in house allocation problems. *Econometrica*, 66 (3):689–701, 1998. ISSN 00129682, 14680262. URL http://www.jstor.org/stable/2998580.

A. Abizada and S. Chen. House allocation when availability of houses may change unexpectedly. *Mathematical Social Sciences*, 81:29 – 37, 2016. ISSN 0165-4896. doi: http://dx.doi.org/10.1016/j.mathsocsci.2016.03.002. URL http://www.sciencedirect.com/science/article/pii/S0165489616000226.

H. Aziz and J. Mestre. Parametrized algorithms for random serial dictatorship. *Mathematical Social Sciences*, 72:1 – 6, 2014. ISSN 0165-4896. doi: http://dx.doi.org/10.1016/j.mathsocsci.2014.07.002. URL http://www.sciencedirect.com/science/article/pii/S0165489614000584.

F. Brandt, V. Conitzer, U. Endriss, J. Lang, and A.D. Procaccia. *Handbook of Computational Social Choice*. Cambridge University Press, New York, NY, USA, 1st edition, 2016. ISBN 1107060435, 9781107060432.

J. Carlier. *Problèmes d'ordonnancements à contraintes de ressources: algorithmes et complexité. Thèse d'état.* Université Paris 6, 1984.

J. Carlier and A.H.G. Rinnooy Kan. Scheduling subject to nonrenewable-resource constraints. *Operations Research Letters*, 1(2):52–55, April 1982. ISSN 0167-6377. doi: 10.1016/0167-6377(82)90045-1. URL http://dx.doi.org/10.1016/0167-6377(82)90045-1.

K. Cechlárová and T. Fleiner. Pareto optimal matchings with lower quotas. *Mathematical Social Sciences*, 88:3 – 10, 2017. ISSN 0165-4896. doi: http://dx.doi.org/10.1016/j.mathsocsci.2017.03.007. URL `http://www.sciencedirect.com/science/article/pii/S0165489617300641`.

K. Cechlárová, P. Eirinakis, T. Fleiner, D. Magos, D. Manlove, I. Mourtos, E. Ocelźáková, and B. Rastegari. Pareto optimal matchings in many-to-many markets with ties. *Theory of Computing Systems*, 59(4):700–721, November 2016. ISSN 1432-4350. doi: 10.1007/s00224-016-9677-1. URL `https://doi.org/10.1007/s00224-016-9677-1`.

X. Chen, X. Hu, T.-Y. Liu, W. Ma, T. Qin, P. Tang, C. Wang, and B. Zheng. Efficient mechanism design for online scheduling. *Journal of Artificial Intelligence Research*, 56(1):429–461, May 2016. ISSN 1076-9757. URL `http://dl.acm.org/citation.cfm?id=3013589.3013601`.

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

S. Dughmi and A. Ghosh. Truthful assignment without money. In *Proceedings of the 11th ACM conference on Electronic commerce*, pages 325–334. ACM, 2010.

C. Edirisinghe and D. Atkins. Lower bounding inventory allocations for risk pooling in two-echelon supply chains. *International Journal of Production Economics*, 187:159–167, 2017. ISSN 0925-5273. doi: https://doi.org/10.1016/j.ijpe.2017.02.015. URL `http://www.sciencedirect.com/science/article/pii/S0925527317300543`.

P. Egri and J. Váncza. Channel coordination with the newsvendor model using asymmetric information. *International Journal of Production Economics*, 135(1):491–499, 2012. ISSN 0925-5273. doi: https://doi.org/10.1016/j.ijpe.2011.08.028. URL `http://www.sciencedirect.com/science/article/pii/S0925527311003823`. Advances in Optimization and Design of Supply Chains.

P. Egri and J. Váncza. A distributed coordination mechanism for supply networks with asymmetric information. *European Journal of Operational Research*, 226(3):452–460, 2013. ISSN 0377-2217. doi: http://dx.doi.org/10.1016/j.ejor.2012.11.036. URL `http://www.sciencedirect.com/science/article/pii/S0377221712008880`.

G. Fandel and J. Trockel. Optimal lot sizing in a non-cooperative material manager – controller game. *International Journal of Production Economics*, 133(1):256 – 261, 2011. ISSN 0925-5273. doi: https://doi.org/10.1016/j.ijpe.2010.12.008. URL `http://www.sciencedirect.com/science/article/pii/S0925527310004779`. Leading Edge of Inventory Research.

E.R. Gafarov, A.A. Lazarev, and F. Werner. Single machine scheduling problems with financial resource constraints: Some complexity results and properties. *Mathematical Social Sciences*, 62(1):7 – 13, 2011. ISSN 0165-4896. doi: http://dx.doi.org/10.1016/j.mathsocsci.2011.04.004. URL `http://www.sciencedirect.com/science/article/pii/S0165489611000321`.

Y. Giannakopoulos, E. Koutsoupias, and M. Kyropoulou. The anarchy of scheduling without money. In Martin Gairing and Rahul Savani, editors, *Algorithmic Game Theory: 9th International Symposium, SAGT 2016, Liverpool, UK, September 19–21, 2016, Proceedings*, pages 302–314, Berlin, Heidelberg, 2016. Springer Berlin Heidelberg. ISBN 978-3-662-53354-3. doi: 10.1007/978-3-662-53354-3_24. URL `https://doi.org/10.1007/978-3-662-53354-3_24`.

A. Grigoriev, M. Holthuijsen, and J. van de Klundert. Basic scheduling problems with raw material constraints. *Naval Research of Logistics*, 52:527–553, 2005. doi: 10.1002/nav.20095.

P. Györgyi and T. Kis. Reductions between scheduling problems with non-renewable resources and knapsack problems. *Theoretical Computer Science*, 565:63–76, 2015a. doi: 10.1016/j.tcs.2014.11.007.

P. Györgyi and T. Kis. Approximability of scheduling problems with resource consuming jobs. *Annals of Operations Research*, 235(1):319–336, 2015b. ISSN 0254-5330. doi: 10.1007/s10479-015-1993-3.

P. Györgyi and T. Kis. Approximation schemes for parallel machine scheduling with non-renewable resources. *European Journal of Operational Research*, 258(1):113 – 123, 2017. ISSN 0377-2217. doi: http://dx.doi.org/10.1016/j.ejor.2016.09.007. URL `http://www.sciencedirect.com/science/article/pii/S0377221716307433`.

B. Heydenreich, R. Müller, and M. Uetz. Games and mechanism design in machine scheduling – an introduction. *Production and Operations Management*, 16(4):437–454, 2007. ISSN 1937-5956. doi: 10.1111/j.1937-5956.2007.tb00271.x. URL `http://dx.doi.org/10.1111/j.1937-5956.2007.tb00271.x`.

I. Kash, A.D. Procaccia, and N. Shah. No agent left behind: Dynamic fair division of multiple resources. *Journal of Artificial Intelligence Research*, 51:579–603, 2014.

T. Kis. Approximability of total weighted completion time with resource consuming jobs. *Operations Research Letters*, 43(6):595–598, 2015. doi: 10.1016/j.orl.2015.09.004.

P. Krysta, O. Telelis, and C. Ventre. Mechanisms for multi-unit combinatorial auctions with a few distinct goods. *Journal of Artificial Intelligence Research*, 53(1):721–744, May 2015. ISSN 1076-9757. URL `http://dl.acm.org/citation.cfm?id=2831071.2831087`.

R. Kurata, N. Hamada, A. Iwasaki, and M. Yokoo. Controlled school choice with soft bounds and overlapping types. *Journal of Artificial Intelligence Research*, 58:153–184, 2017.

D. Manlove. *Algorithmics Of Matching Under Preferences*. Theoretical computer science. World Scientific Publishing, 2013. URL `http://eprints.gla.ac.uk/79820/`.

N. Nisan and A. Ronen. Algorithmic mechanism design. *Games and Economic Behavior*, 35(1):166 – 196, 2001. ISSN 0899-8256. doi: http://dx.doi.org/10.1006/game.1999.0790. URL `http://www.sciencedirect.com/science/article/pii/S089982569990790X`.

V. Robu, E.H. Gerding, S. Stein, D.C. Parkes, A. Rogers, and N.R. Jennings. An online mechanism for multi-unit demand and its application to plug-in hybrid electric vehicle charging. *Journal of Artificial Intelligence Research*, 48(1):175–230, October 2013. ISSN 1076-9757. URL `http://dl.acm.org/citation.cfm?id=2591248.2591253`.

J. Rothe. *Economics and Computation: An Introduction to Algorithmic Game Theory, Computational Social Choice, and Fair Division*. Springer Publishing Company, Incorporated, 1st edition, 2015. ISBN 3662479036, 9783662479032.

J. Sternatz. The joint line balancing and material supply problem. *International Journal of Production Economics*, 159:304–318, 2015. ISSN 0925-5273. doi: https://doi.org/10.1016/j.ijpe.2014.07.022. URL `http://www.sciencedirect.com/science/article/pii/S0925527314002394`.

L. Tiacci. Simultaneous balancing and buffer allocation decisions for the design of mixed-model assembly lines with parallel workstations and stochastic task times. *International Journal of Production Economics*, 162:201–215, 2015. ISSN 0925-5273. doi: https://doi.org/10.1016/j.ijpe.2015.01.022. URL `http://www.sciencedirect.com/science/article/pii/S0925527315000341`.

J. Váncza and A. Márkus. An agent model for incentive-based production scheduling. *Computers in Industry*, 43(2):173 – 187, 2000. ISSN 0166-3615. doi: http://dx.doi.org/10.1016/S0166-3615(00)00066-X. URL `http://www.sciencedirect.com/science/article/pii/S016636150000066X`.

A Modification of the Carlier – Rinnooy Kan algorithm

In Carlier and Rinnooy Kan [1982], the following inequalities can be found (considering only a single resource): $\forall \ell : \sum_{j \in J} \{ a_j \mid \gamma \leq \gamma_{\ell j} \} \leq B(u_\ell)$, and "for fixed ℓ, the smallest value γ^*_ℓ for which [the inequality] is satisfied can be found by a *median* finding procedure [...]". However, such smallest value γ^*_ℓ

may not exist. Consider the following simple example with a single job j and one resource only. There are two supplies at times $u_1 = 0$, and $u_2 = 1$ with supplied quantities $b_{1,1} = 1$ and $b_{1,2} = 1$, respectively, and demand $a_j = 2$. Hence, the cumulative supplies are $B(u_1) = 1$ and $B(u_2) = 2$, respectively, and job j can start only at u_2. Then for $\ell = 1$, with $\gamma = \gamma_{1,1}$ the inequality does not hold, but for any $\epsilon > 0$, with $\gamma = \gamma_{1,1} + \epsilon$ the inequality is satisfied, since the left-hand-side is 0.

Thus we use $\forall \rho, \ell : \sum_{j \in J}\{\, a_{\rho j} \,|\, \gamma < \gamma_{\ell j} \,\} \leq B_\rho(u_{\ell-1})$ instead. We now show that if these inequalities are satisfied, then the resulted maximal tardiness cannot be greater than γ in an optimal schedule. Let us indirectly assume that for an optimal μ allocation there exists a project p with $T_p^{(\mu)} > \gamma$. This means that $\exists j^* \in J_p : f_{j^*}(e_{j^*}^{(\mu)}) > \gamma$. Let us consider a chain $(j_1, \ldots, j_{l_{\max}} = j^*)$, where $(j_l, j_{l+1}) \in A_p$ and $e_{j_l}^{(\mu)} = s_{j_{l+1}}^{(\mu)}$, but there exists no job k with $(k, j_1) \in A_p$ and $e_k^{(\mu)} = s_{j_1}^{(\mu)}$. Then $s_{j_1}^{(\mu)} = u_\ell$ must hold for some ℓ. But then, by definition, $\gamma_{\ell j_1} = f_{j^*}(e_{j^*}^{(\mu)}) > \gamma$, and neither the precedence constraints (by the choice of j_1), nor resource availability (since $\sum_{j \in J}\{\, a_{\rho j} \,|\, \gamma < \gamma_{\ell j} \,\} \leq B_\rho(u_{\ell-1})$ by assumption) blocks j_1, which contradicts the assumption that j_1 starts at the earliest start time permitted by the resource and the precedence constraints.