Noname manuscript No.
(will be inserted by the editor)

Minimizing total weighted completion time on a single machine subject to non-renewable resource constraints

Péter Györgyi · Tamás Kis

Received: date / Accepted: date

Abstract In this paper we describe new complexity results, and approximation algorithms for single machine scheduling problems with non-renewable resource constraints and the total weighted completion time objective. This problem is hardly studied in the literature, and most of the published results establish the computational complexity of various special cases with different objective functions. In this paper we discuss some polynomially solvable special cases and also show that under very strong assumptions, like the processing time, the resource consumption and the weight is the same for each job, minimizing the total weighted completion time is still NP-hard. In addition, we also propose a 2-approximation algorithm for this variant, and a PTAS for the case when the processing time equals the weight for each job, while the resource consumptions are arbitrary.

Keywords single machine scheduling, non-renewable resources, approximation algorithms.

1 Introduction

Non-renewable resources, like raw material, energy, or money, are used in all sectors of production, and depending on the stocking policy, they have varying impact on the preparation of daily and weekly production schedules. Consider for instance the preparation of the weekly schedule of a production line, where some of the raw materials built into the products arrive over the week, and the supplies constrain what and when can be produced. Of course, if all the purchased items were on stock right at the beginning of the week, then the supply arriving during the week would not influence the scheduling decisions, but the drawback is that larger stocks should be kept which incurs additional costs.

In this paper we consider single machine scheduling problems with one additional non-renewable resource. The non-renewable resource has an initial stock, and some additional supplies in the future with known supply dates and quantities. A job can only be started if the inventory level of the resource is at least as much as the quantity required by the job. When the job is started, the inventory level is decreased by the required quantity. Therefore, when determining the schedule, one has to take into account not only the initial stock level, but also the future supplies. This is an extra constraint beside e.g., job release dates, or sequence dependent setup times.

More formally, in all problems studied in this paper, there is a single machine, a non-renewable resource, and a finite set of jobs $\mathcal{J}$. Each job $j \in \mathcal{J}$ has a processing time $p_j > 0$, a weight $w_j \geq 0$, and a resource requirement $a_j \geq 0$. The resource has an initial supply $\tilde{b}_1$ available at time $u_1 = 0$, and additional supplies $\tilde{b}_\ell$ at supply dates u_ℓ for $\ell = 2, \ldots, q$. For convenience, we also define $u_{q+1} = +\infty$. We assume that the supplies are indexed in increasing u_ℓ order, i.e., $u_\ell < u_{\ell+1}$ for $\ell = 1, \ldots, q-1$. Let S be a schedule specifying a start time for each job. It is feasible if (i) the jobs do not overlap in time, and (ii) for each $\ell = 1, \ldots, q$, $\sum_{j:S_j<u_{\ell+1}} a_j \leq \sum_{\ell'=1}^{\ell} \tilde{b}_\ell$, i.e., the supply arriving up to u_ℓ covers the demands of those jobs starting before $u_{\ell+1}$. The objective function is the weighted sum of job completion times, i.e., a feasible

P. Györgyi
Institute for Computer Science and Control, Hungarian Academy of Sciences, H1111 Budapest, Kende str. 13–17, Hungary
T. Kis
Institute for Computer Science and Control, Hungarian Academy of Sciences, H1111 Budapest, Kende str. 13–17, Hungary
Tel.: +36 1 2796156; Fax: +36 1 4667503
E-mail: peter.gyorgyi@sztaki.mta.hu, tamas.kis@sztaki.mta.hu

schedule of minimum $\sum_{j\in\mathcal{J}} w_jC_j$ value is sought, where $C_j = S_j + p_j$. We mention that a feasible schedule exists only if $\sum_{j\in\mathcal{J}} a_j \le \sum_{\ell=1}^{q} \tilde{b}_\ell$, and more resources are not needed. In fact, without loss of generality we may assume that

i) $\sum_{j\in\mathcal{J}} a_j = \sum_{\ell=1}^{q} \tilde{b}_\ell$, and
ii) $\tilde{b}_q > 0$, i.e., at least one job must start not before u_q.

In the standard $\alpha|\beta|\gamma$ notation of Graham et al. [6], we will indicate in the β field by $nr = 1$ that the number of non-renewable resources is 1. In addition, we will constrain the number of supply dates to a constant by $q = const$. We will use a number of other constraints which are standard in the scheduling literature.

There are only sporadic results on this problem. Carlier [4] has established that $1|nr = 1|\sum w_jC_j$ is NP-hard in the strong sense, which was also established in Gafarov et al. [5]. However, the problem remains NP-hard in the weak sense if $q = 2$ (two supplies), see Kis [15]. In [15], an FPTAS is devised for the special case $1|nr = 1, q = 2|\sum w_jC_j$. Moreover, Gafarov et al. [5] study a variant of this problem, where each job has processing time 1, and there are n supplies such that $u_\ell = \ell M$, and $\tilde{b}_\ell = M$ for $\ell = 1,\dots,n$, where $M = \sum_{j\in\mathcal{J}} a_j/n$ is an integer number, and $n = |\mathcal{J}|$. Without the non-renewable resource constraint, the problem $1||\sum w_jC_j$ can be solved optimally in polynomial time by scheduling the jobs in non-increasing w_j/p_j order, a classical result of Smith [18].

In this paper we establish new complexity and approximability results for special cases of $1|nr = 1|\sum w_jC_j$. The special cases are obtained by imposing constraints on the parameters of the jobs. For instance, the constraint $p_j = w_j$ means that for each job, its processing time equals its weight, $a_j = \bar{a}$ indicates that all the jobs have a common resource requirement, whereas $p_j = 1$ or $p_j = \bar{p}$ restricts the processing time of each job to 1 or to some other common constant value. The new results are summarized in Table 1. As we can see, 5 special cases can be solved in polynomial time by list scheduling, we identify 3 new NP-hard variants, and propose approximation algorithms in two cases. We emphasize that the 2-approximation algorithm is merely list scheduling using the LPT order, but the analysis of the algorithm is tricky. On the other hand, the polynomial time approximation scheme for $1|nr = 1, w_j = p_j, q = const.|\sum p_jC_j$ is rather involved, and the underlying analysis needs new ideas which may be used in the analysis of other problems as well.

The structure of the paper: in Section 2 we overview the related literature. In Section 3 we generalize list scheduling to our problem, and discuss special cases that can be solved optimally with this method. In Section 4 we establish the NP-hardness of $1|nr = 1, p_j = 1, a_j = w_j|\sum w_jC_j$. In Section 5 we present complexity results and a 2-approximation algorithm for the special case with $p_j = a_j = w_j$. Finally, in Section 6 we devise a PTAS for $1|nr = 1, p_j = w_j, q = const.|\sum w_jC_j$.

2 Literature review

Machine scheduling problems with non-renewable resources have been introduced by Carlier [4], and Slowinski [17]. In [4], the computational complexity of several variants with a single machine is established. Slowinski [17] considers a parallel machine problem with preemptive jobs, and with a single non-renewable resource which has an initial stock and some additional supplies. It is assumed that the rate of consuming the non-renewable resource is constant during the execution of the jobs. These assumptions led to a polynomial time algorithm for minimizing the makespan. Toker et al. [19] prove that the single machine scheduling problem with a single non-renewable resource and the makespan objective reduces to the 2-machine flow shop problem provided that the single non-renewable resource has a unit supply in every time period. In [20], [7] and [5] the complexity of several variants with the minimum makespan objective is studied, and some constant ratio approximation algorithms are developed in [7]. In [8] [9], [11], [10], [12] the approximability of several variant of the makepsan minimization problem on single and parallel machines is established, and in [13] a branch-and-cut algorithm for minimizing the maximum lateness is devised and evaluated.

A related model with resource producing and resource consuming jobs is studied in [2], [3], [14] with the objective of minimizing the maximum inventory level needed to complete all the jobs (there are no external supplies over the scheduling horizon). In Kellerer et al. [14] contant ratio approximation algorithms are developed, and in Briskorn et al. [2] the complexity of several variants is established. In Briskorn et al. [3] an exact method is devsied. Morsy and Pesch [16] design a 2-approximation algorithm to minimize the total weighted completion time in a special case of the scheduling problem with producer and consumer jobs.

Table 1 New complexity and approximability results.

#Res. nr	#Supp. q	Restriction	Objective function	Result
1	$*$	$p_j = a_j = \bar{a}$	$\sum w_j C_j$	polytime (decr. w_j ord.)
1	$*$	$p_j = w_j = 1$	$\sum C_j$	polytime (incr. a_j ord.)
1	$*$	$a_j = w_j = 1$	$\sum C_j$	polytime (incr. p_j ord.)
1	$*$	$w_j = \bar{w}, p_j = a_j$	$\sum \bar{w} C_j$	polytime (incr. p_j ord.)
1	$*$	$a_j = \bar{a}, p_j = w_j$	$\sum p_j C_j$	polytime (decr. p_j ord.)
1	2	$w_j = p_j = a_j$	$\sum p_j C_j$	weakly NP-hard
1	$*$	$w_j = p_j = a_j$	$\sum p_j C_j$	strongly NP-hard
1	2	$p_j = 1$, $w_j = a_j$	$\sum w_j C_j$	weakly NP-hard
1	$*$	$w_j = p_j = a_j$	$\sum p_j C_j$	2-approx algo (LPT rule)
1	*const.*	$w_j = p_j$	$\sum p_j C_j$	PTAS

"$*$" stands for "arbitrary"
"decr. w_j ord." means decreasing (non-increasing) w_j order.
"incr. p_j ord." is equivalent to SPT rule.
"decr. p_j ord." is equivalent to LPT rule.
"2-approx algo" means "polytime time approximation algorithm with relative error 2".
"PTAS" stands for "polynomial time approximation scheme".

3 List scheduling

In this section we discuss polynomially solvable special cases of $1|nr = 1|\sum w_j C_j$. All the algorithms presented below are based on the following scheme:

1. Sort the jobs according to some total ordering relation. Let $L = (j_1, \ldots, j_n)$ be the sequence obtained. Let $t := 0$, $\ell := 1$, and $r := b_1$.
2. For $i = 1$ to n do
3. While $a_{j_i} > r$ repeat let $\ell := \ell + 1$, $t := \max\{t, u_\ell\}$, and $r := r + \tilde{b}_\ell$. End-while.
4. Schedule j_i at time t. That is, set $S_{j_i} := t$, and then $t := t + p_{j_i}$, $r := r - a_{j_i}$.
5. End-for
6. Output S.

In the above algorithm, t represents the time when the next job may be scheduled, and r the resource level before scheduling it. In Step 3, t and r is reset if the resource available after scheduling the previous jobs is not enough to schedule j_i. Notice that in such a case, the supply of more than one period may be needed to increase the available quantity of the resource sufficiently.

The above simple algorithm is a generalization of the well-known algorithm which schedules the jobs in some given order without interruptions, see e.g., [1].

Now we present the already announced special cases, which differ in the restrictions on the various job parameters:

$1|nr = 1, p_j = a_j = \bar{a}|\sum w_j C_j$: Schedule the jobs in non-increasing w_j order.
$1|nr = 1, p_j = w_j = 1|\sum w_j C_j$: Schedule the jobs in non-decreasing a_j order.
$1|nr = 1, a_j = w_j = 1|\sum w_j C_j$: Schedule the jobs in SPT[1] order.
$1|nr = 1, w_j = \bar{w},\ p_j = a_j|\sum w_j C_j$: Schedule the jobs in SPT order.
$1|nr = 1, a_j = \bar{a},\ p_j = w_j|\sum w_j C_j$: Schedule the jobs in LPT[2] order.

The proof of optimality is left to the reader, except in the last case:

Theorem 1 *The LPT schedule yields an optimal solution for* $1|nr = 1, a_j = \bar{a}, p_j = w_j|\sum w_j C_j$.

Proof Let S^* be an optimal schedule, and $S^*_j + p_j = C^*_j$ for each job j in which the number of job pairs violating the LPT order is the smallest. Suppose that there are at least two jobs that are not in LPT order. Consider the first two such consecutive jobs, say j_1 and j_2, where j_1 is scheduled before j_2, and $p_{j_1} + K = p_{j_2}$ for some $K > 0$. Let S' be the schedule where we swap the order of j_1 and j_2. We distinguish two cases.

If $C^*_{j_1} = S^*_{j_2}$, then $S'_{j_1} = S^*_{j_1} + p_{j_2}$, $S'_{j_2} = S^*_{j_1}$, and $S'_j = S^*_j$ for all $j \notin \{j_1, j_2\}$. It is easy to verify that $w_{j_1}(S^*_{j_1} + p_{j_1}) + w_{j_2}(S^*_{j_2} + p_{j_2}) = w_{j_2}(S'_{j_2} + p_{j_2}) + w_{j_1}(S'_{j_1} + p_{j_1})$, and the objective function does not change. Since S' is feasible, as each job has the same resource requirement, we reached a contradiction with the choice of S^*.

Now suppose $C^*_{j_1} < S^*_{j_2}$. Hence, there is an ℓ such that $S^*_{j_2} = u_\ell$. Note that we have $S'_{j_2} = S^*_{j_1}$, $S'_{j_1} = \max\{C'_{j_2}, u_\ell\}$. Further on we have $S'_j = S^*_j$ for each job j with $S^*_j < S^*_{j_1}$, and $S'_j \le S^*_j$ for each job j with $S^*_j \ge S^*_{j_2}$. Notice that only the start time of job j_1 increases after swapping job j_1 and job j_2. To reach a contradiction with the choice of S^*, it is enough to

[1] non-decreasing in p_j [2] non-increasing in p_j

prove that $w_{j_1}C^*_{j_1}+w_{j_2}C^*_{j_2} \geq w_{j_1}C'_{j_1}+w_{j_2}C'_{j_2}$. Suppose that we have $u_\ell = S^*_{j_1}+p_{j_1}+L$ where $L>0$. We have

$$\begin{aligned} w_{j_1}C^*_{j_1}+w_{j_2}C^*_{j_2} &= p_{j_1}(S^*_{j_1}+p_{j_1})+ \\ &\quad p_{j_2}\cdot((S^*_{j_1}+p_{j_1}+L)+p_{j_2}) \\ &= p_{j_1}S^*_{j_1}+p^2_{j_1}+(p_{j_1}+K)S^*_{j_1}+ \\ &\quad (p_{j_1}+K)(p_{j_1}+L)+(p_{j_1}+K)^2, \end{aligned}$$

and

$$\begin{aligned} w_{j_1}C'_{j_1}+w_{j_2}C'_{j_2} &= p_{j_1}(S'_{j_1}+p_{j_1})+ \\ &\quad (p_{j_1}+K)(S^*_{j_1}+p_{j_1}+K) \\ &= p_{j_1}\max\{C'_{j_2},u_\ell\}+ \\ &\quad p^2_{j_1}+(p_{j_1}+K)S^*_{j_1}+(p_{j_1}+K)^2. \end{aligned}$$

Thus, $w_{j_1}C^*_{j_1}+w_{j_2}C^*_{j_2}-(w_{j_1}C'_{j_1}+w_{j_2}C'_{j_2}) = p_{j_1}S^*_{j_1}+(p_{j_1}+K)\cdot(p_{j_1}+L)-p_{j_1}\max\{C'_{j_2},u_\ell\}$. Since $\max\{C'_{j_2},u_\ell\} = \max\{S^*_{j_1}+p_{j_1}+K, S^*_{j_1}+p_{j_1}+L\}$, thus $w_{j_1}C^*_{j_1}+w_{j_2}C^*_{j_2} \geq w_{j_1}C'_{j_1}+w_{j_2}C'_{j_2}$ follows. □

4 Problem $1|nr=1, p_j=1, a_j=w_j|\sum w_jC_j$

Theorem 2 *The problem* $1|nr=1,p_j=1,a_j=w_j|\sum w_jC_j$ *is NP-hard.*

Proof We reduce the NP-hard PARTITION problem to our scheduling problem. An instance of the former problem is given by a natural number n, and the sizes of n items, $s_1,\ldots,s_n$, which are non-negative integer numbers. One has to decide whether the items can be partitioned into two subsets, Q_1 and Q_2, such that $\sum_{i\in Q_1}s_i = \sum_{i\in Q_2}s_i$. Since all item sizes are integer numbers, the answer is 'NO', unless $\sum_{i=1}^n s_i = 2A$ for some integer A. Therefore, we assume that $\sum_{i=1}^n s_i$ is an even integer, and let $A := \sum_{i=1}^n s_i/2$. Let I be an instance of PARTITION, the corresponding instance I' of $1|nr=1,p_j=1,a_j=w_j|\sum w_jC_j$ consists of n jobs, and for each item j, the corresponding job has a processing time $p_j=1$, and $w_j := s_j$ and $a_j := s_j$. In addition, there is a single resource with an initial stock of $\tilde{b}_1 := A$, available at time $u_1 := 0$, and with one more supply $\tilde{b}_2 := A$ at time $u_2 := n^2A^2$.

We claim that I has a 'YES' answer if and only if I' has a feasible schedule of objective function value at most n^2A^3+2nA. First suppose that I has a partitioning of the items Q_1, Q_2 of equal size. Schedule the jobs corresponding to the items in Q_1 from time 0 on consecutively in decreasing w_j order, and those in Q_2 from u_2 consecutively in decreasing w_j order. This schedule is clearly feasible. Suppose $Q_1=\{j_1,\ldots,j_k\}$, and $w_{j_i} \geq w_{j_{i+1}}$ for $i=1,\ldots,k-1$, and $Q_2=\{j_{k+1},\ldots,j_n\}$, and $w_{j_i} \geq w_{j_{i+1}}$ for $i=k+1,\ldots,n-1$. Then we compute

$$\sum_j w_jC_j = \sum_{i=1}^k iw_{j_i} + \sum_{i=k+1}^n (n^2A^2+i-k)w_{j_i} < n^2A^3+2nA.$$

Conversely, suppose the scheduling problem admits a feasible schedule S of objective function value at most n^2A^3+2nA. Let $C_j := S_j+1$ for each job j. Let $Q_1 = \{j \mid S_j < u_2\}$ and $Q_2 = \{j \mid S_j \geq u_2\}$. Since S is feasible, the total resource consumption of those jobs in Q_1 is at most A. Indirectly, suppose it is less than A. Then the total weight of those jobs in Q_2 is at least $A+1$. But then we have

$$\sum_j w_jC_j \geq \sum_{j\in Q_2} n^2A^2w_j \geq n^2A^3+n^2A^2 > n^2A^3+2nA,$$

which is a contradiction.

Finally, notice that the transformation is of polynomial time complexity, which shows that there is a polynomial reduction from PARTITION to a decision version of the scheduling problem $1|nr=1,p_j=1,a_j=w_j|\sum w_jC_j$. □

5 Problem $1|nr=1, p_j=a_j=w_j|\sum w_jC_j$

We start this section by providing a non-trivial expression for the objective function value of an optimal schedule under the condition $p_j=w_j$ for every job j.

Let S be any feasible schedule for the problem and let $C_j = S_j+p_j$ be the completion time of job j in S. Let H_ℓ denote the length of the idle period, if any, in schedule S in the interval $[u_\ell, u_{\ell+1}]$ and let $G_\ell = \sum_{\nu=1}^{\ell} H_\nu$ be the total idle time until $u_{\ell+1}$. Let P_ℓ denote the total working time (when the machine is not idle) in $[u_\ell, u_{\ell+1}]$, noting that $u_\ell = \sum_{\nu=1}^{\ell-1}P_\nu + G_{\ell-1}$. See Figure 1 (a) for an illustration. Using the new notation, we can express the objective function value of S as follows:

Lemma 1 *If* $p_j=w_j$ *for each job* j*, then the objective function value of any feasible schedule* S *can be expressed as*

$$\begin{aligned} \sum_j p_jC_j &= \sum_{j\leq k}p_jp_k + \sum_{\ell=2}^{q} G_{\ell-1}P_\ell \\ &= \sum_{j\leq k}p_jp_k + \sum_{\ell=2}^{q} H_{\ell-1}(P_\ell+P_{\ell+1}+\ldots+P_q). \end{aligned} \tag{1}$$

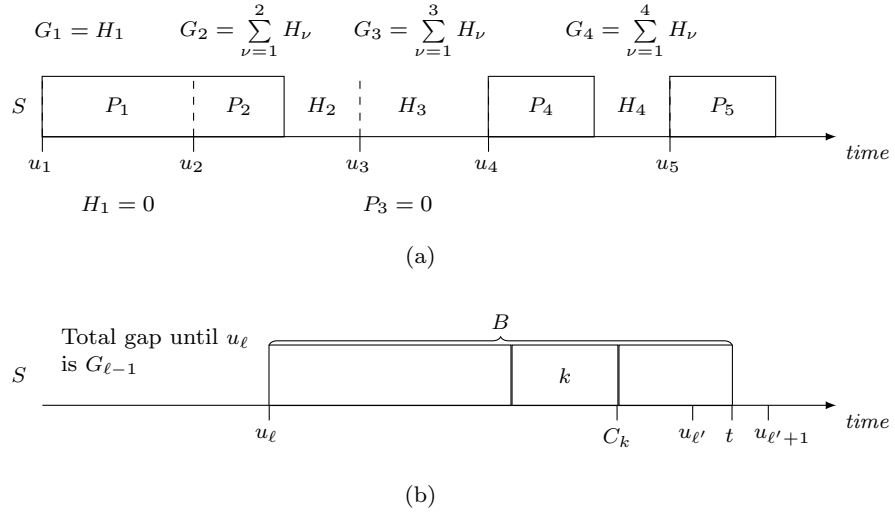

Fig. 1 (a) The new notations (G_ℓ, H_ℓ and P_ℓ); (b) proof of Lemma 1.

Proof Consider any working period $B = [u_\ell, t]$ in the schedule S, that is, the machine is idle right before u_ℓ and right after t, and is working contiguously throughout B. Suppose $t \in (u_{\ell'}, u_{\ell'+1}]$, where $\ell' \geq \ell$. Let k be an arbitrary job that is processed in B, see Figure 1 (b). We have $C_k = \sum_{C_j \leq C_k} p_j + G_{\ell-1}$, thus the total weighted completion time of the jobs processed in B is

$$\sum_{k:C_k \in B} p_k \left(\sum_{C_j \leq C_k} p_j + G_{\ell-1} \right) =$$

$$\sum_{k:C_k \in B} p_k \sum_{C_j \leq C_k} p_j + G_{\ell-1} \sum_{\nu=\ell}^{\ell'} P_\nu =$$

$$\sum_{k:C_k \in B} p_k \sum_{C_j \leq C_k} p_j + \sum_{\nu=\ell}^{\ell'} G_{\nu-1} P_\nu,$$

where the first equation follows from $\sum_{k:C_k \in B} p_k = \sum_{\nu=\ell}^{\ell'} P_\mu$, and the second from $G_\nu = G_{\ell-1}$ for each $\ell \leq \mu < \ell'$, since the machine is not idle in the interval B. Since the schedule can be partitioned into working and idle periods, we derive

$$\sum_j w_j C_j = \sum_{j \leq k} p_j p_k + \sum_{\ell=2}^{q} G_{\ell-1} P_\ell.$$

Finally, the second equation of the statement of the lemma can be derived by using the definition of G_ℓ and by rearranging terms. □

Theorem 3 *The problem* $1|nr = 1, q = 2, p_j = a_j = w_j| \sum w_j C_j$ *is weakly NP-hard, and* $1|nr = 1, p_j = a_j = w_j| \sum w_j C_j$ *is strongly NP-hard.*

Proof For proving the weak NP-hardness of $1|nr = 1, q = 2, p_j = a_j = w_j| \sum w_j C_j$ we reduce the PARTITION problem to this scheduling problem. Recall that an instance of PARTITION is given by a positive integer n, and n non-negative integer numbers $s_1, \ldots, s_n$, that represent the respective size of n distinct items. One has to decide whether the items can be partitioned into two subsets, Q_1 and Q_2, such that $\sum_{i \in Q_1} s_i = \sum_{i \in Q_2} s_i$. Since the item sizes are integer numbers, the answer is 'NO', unless $\sum_{i=1}^{n} s_i = 2A$ for some integer A. Therefore, we assume that $\sum_{i=1}^{n} s_i = 2A$ in any instance of PARTITION, and the question can be equivalently stated as if there exists a subset Q of items with $\sum_{i \in Q} s_i = A$. Now, the corresponding instance of $1|nr = 1, q = 2, p_j = a_j = w_j| \sum w_j C_j$ consists of n jobs, one job foe each item, and $p_i = a_i = w_i = s_i$ for each item $i = 1, \ldots, n$. There are two supplies, one at $u_1 = 0$ and the supplied quantity from the single resource is A, and another at $u_2 = A$ with supplied quantity A. We claim that the the PARTITION problem instance has a solution if and only if the corresponding scheduling problem instance has a feasible solution of value at most $\sum_{j \leq k} p_j p_k$. Using Lemma 1, the the latter holds if and only if the schedule has no idle time. So, it suffices to prove that the PARTITION problem instance has a solution if and only if the corresponding scheduling problem instance admits a feasible schedule without any idle time. First suppose that the PARTITION problem instance has a 'yes' answer, i.e., there is a subset Q of items with $\sum_{i \in Q} s_i = A$. Schedule the corresponding jobs contiguously in any order in the interval $[0, A]$. Since $p_j = a_j$, and the supply at $u_1 = 0$ is A, this is feasible. Now, schedule the remaining jobs without idle times from $u_2 = A$. The result is a feasible

schedule without idle times. Conversely, suppose there is a feasible schedule without idle times. Then the machine is working throughout the interval $[0, A]$. Since the supply at $u_1 = 0$ is A, the total processing time of the jobs starting before $u_2 = A$ is A. Let the set Q consist of the items corresponding to these jobs. This yields a feasible solution for the PARTITION problem instance.

For proving the strong NP-hardness of $1|nr = 1, p_j = a_j = w_j|\sum w_j C_j$ we reduce the 3-PARTITION problem to this scheduling problem. Recall that an instance of 3-PARTITION consists of an positive integer t, and $3t$ items, each having a size s_i, $i \in \{1, \ldots, 3t\}$, where the item sizes are bounded by polynomial in the input length. It is assumed that $\sum_{i=1}^{3t} s_i$ is divisible by t, and $B/4 < s_i < B/2$ for each i, where $B = \sum_{i=1}^{3t} s_i/t$. The question is whether the set of items can be partitioned into t groups $Q_1, \ldots, Q_t$ such that $\sum_{i \in Q_\ell} s_i = B$ for $\ell = 1, \ldots, t$. The corresponding instance of the scheduling problem $1|nr = 1, p_j = a_j = w_j|\sum w_j C_j$ has $3t$ jobs corresponding to the $3t$ items with $p_i = a_i = w_i = s_i$, and $q = t$ supplies at supply dates $u_\ell = (\ell - 1)B$ with supplied quantities $b_\ell = B$ for $\ell = 1, \ldots, q$. The rest of the proof goes along the same lines as in the first part, i.e., we argue that 3-PARTITION has a feasible solution if and only the the corresponding scheduling problem instance has a solution of objective function value $\sum_{j \le k} p_j p_k$ if and only if there is a feasible schedule without any idle times. □

Theorem 4 *Scheduling the jobs in LPT order is a 2-approximation algorithm for* $1|nr = 1, p_j = a_j = w_j|\sum w_j C_j$.

Proof The main idea of the following proof is that first we transform the problem data such that the resource supplies are deferred until they are used in a selected optimal schedule, and then we bound the approximation ratio of the LPT schedule. Finally, we observe that the LPT order yields at least as good a schedule with the original problem data as the same job order for the modified problema data.

Let I be any instance of the scheduling problem, and fix an optimal schedule S^* for I. Let $\mathcal{J}_\ell^*$ be the set of jobs that start in $[u_\ell, u_{\ell+1})$ in S^*. Let I' be a new problem instance derived from I by modifying the supplied quantities (the other problem data does not change): $b'_1 := \sum_{j \in \mathcal{J}_1^*} a_j$ and for each $\ell \ge 2$, $b'_\ell := \sum_{\nu=1}^{\ell} \sum_{j \in \mathcal{J}_\nu^*} a_j - \sum_{\nu=1}^{\ell-1} b'_\nu$.

Claim 1 *I' has the following properties:*

(i) $b'_\ell \ge 0$ *for each* $\ell = 1, \ldots, q$,
(ii) $\sum_{\ell=1}^{q} b'_\ell = \sum_{j=1}^{n} a_j$,
(iii) *S^* is optimal for I',*
(iv) *any ordering of the jobs yields at least as good a schedule for I as for I'.*

Proof The first two claims are straightforward consequences of the definitions, while (iii) and (iv) both follow from the fact that in I' the resource supplies are deferred with respect to I. □

From now on we consider I'.

Let S^{LPT} denote the schedule obtained from the LPT order for problem instance I', and let C_j^{LPT} denote the completion time of job j in this schedule. Let G_ℓ^{LPT} denote the total idle time in S^{LPT} in $[0, u_{\ell+1}]$ and P_ℓ^{LPT} the total working time (when the machine processes a job) in $[u_\ell, u_{\ell+1}]$. We have $u_\ell = \sum_{\nu=1}^{\ell-1} P_\nu + G_{\ell-1}^{LPT}$.

Let us define $\tilde{P}_\ell^{LPT}$ as follows. If the machine is working just before u_ℓ, or idle just after u_ℓ in S^{LPT}, then $\tilde{P}_\ell^{LPT} = 0$; otherwise $\tilde{P}_\ell^{LPT}$ equals the length of the working period starting at u_ℓ until the first idle period in S^{LPT}, see Figure 2. Notice that if the machine is working right before and also right after u_ℓ, then $\tilde{P}_\ell^{LPT} = 0$ by definition.

According to Lemma 1, we can express the total weighted processing time of the LPT schedule as follows:

$$\begin{aligned} \sum_{j \in \mathcal{J}} p_j C_j^{LPT} &= \sum_{j \le k} p_j p_k + \sum_{\ell=2}^{q} G_{\ell-1}^{LPT} P_\ell^{LPT} \\ &= \sum_{j \le k} p_j p_k + \sum_{\ell=2}^{q} G_{\ell-1}^{LPT} \tilde{P}_\ell^{LPT}. \end{aligned} \tag{2}$$

Note that the second equation follows from the fact that if $\tilde{P}_\ell^{LPT} = 0$, then $G_{\ell-1}^{LPT} = G_{\ell'-1}^{LPT}$ for the largest $\ell' < \ell$ with $\tilde{P}_{\ell'}^{LPT} > 0$.

In the next claim we relate (2) to (1). The notations P_ℓ^*, G_ℓ^* and H_ℓ^* refer to P_ℓ, G_ℓ and H_ℓ in case of S^*. Note that $u_\ell = \sum_{\nu=1}^{\ell-1} P_\nu^* + G_{\ell-1}^*$.

Claim 2 *If $\tilde{P}_\ell^{LPT} > 0$, i.e., the machine is idle just before u_ℓ, and a job $j(\ell)$ is started at u_ℓ in S^{LPT}, then*

(i) $\sum_{\nu=1}^{\ell-1} \tilde{P}_\nu^{LPT} + p_{j(\ell)} > \sum_{\nu=1}^{\ell-1} P_\nu^*$ *and* $\sum_{\nu=\ell}^{q} \tilde{P}_\nu^{LPT} < \sum_{\nu=\ell}^{q} P_\nu^* + p_{j(\ell)}$,
(ii) $G_{\ell-1}^{LPT} < G_{\ell-1}^* + p_{j(\ell)}$.

Proof If $\sum_{\nu=1}^{\ell-1} \tilde{P}_\nu^{LPT} + p_{j(\ell)} \le \sum_{\nu=1}^{\ell-1} b_\nu$ were true, then $j(\ell)$ could be scheduled earlier in S^{LPT}. Thus we have $\sum_{\nu=1}^{\ell-1} \tilde{P}_\nu^{LPT} + p_{j(\ell)} > \sum_{\nu=1}^{\ell-1} b_\nu$. Since we have $\sum_{\nu=1}^{\ell-1} P_\nu^* \le \sum_{\nu=1}^{\ell-1} b_\nu$, (i) follows. The second inequality of (i) follows from $\sum_{\nu=1}^{q} \tilde{P}_\nu^{LPT} = \sum_{\nu=1}^{q} P_\nu^*$. Finally, (ii) follows from $\sum_{\nu=1}^{\ell-1} \tilde{P}_\nu^{LPT} + G_{\ell-1}^{LPT} = u_\ell = \sum_{\nu=1}^{\ell-1} P_\nu^* + G_{\ell-1}^*$. □

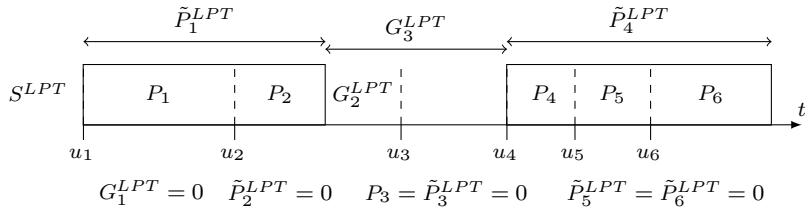

Fig. 2 Notations for the LPT schedule.

Using (2) and Claim 2 (ii), we derive

$$\begin{aligned}\sum_{j\in\mathcal{J}} p_j C_j^{LPT} &\le \sum_{j\le k} p_j p_k + \sum_{\ell=2}^{q} (G^*_{\ell-1} + p_{j(\ell)}) \cdot \tilde{P}_\ell^{LPT} \\ &\le 2 \cdot \sum_{j\le k} p_j p_k + \sum_{\ell=2}^{q} G^*_{\ell-1} \tilde{P}_\ell^{LPT} \\ &= 2 \cdot \sum_{j\le k} p_j p_k + \sum_{\ell=2}^{q} \left(\sum_{\nu=1}^{\ell-1} H^*_\nu \right) \tilde{P}_\ell^{LPT} \\ &= 2 \cdot \sum_{j\le k} p_j p_k + \sum_{\ell=2}^{q} H^*_{\ell-1} \left(\sum_{\mu=\ell}^{q} \tilde{P}_\mu^{LPT} \right),\end{aligned}$$

where the first inequality follows from Claim 2 (ii), the second from the observation that $p_{j(\ell)}$ is multiplied by the total processing time of job $j(\ell)$ and all those jobs following $j(\ell)$ in the LPT order, and the rest is obtained by rearranging terms.

Since $\sum_j p_j C_j^* = \sum_{j\le k} p_j p_k + \sum_{\ell=2}^{q} H^*_{\ell-1} \left(\sum_{\mu=\ell}^{q} P^*_\mu \right)$ (from Lemma 1), it is enough to prove

Claim 3

$$\sum_{\mu=\ell}^{q} \tilde{P}_\mu^{LPT} \le 2 \cdot \sum_{\mu=\ell}^{q} P^*_\mu \quad \forall \ell \ge 2 : \; H^*_{\ell-1} \neq 0.$$

Note that $H^*_{\ell-1} \neq 0$ means the machine is not working before u_ℓ in S^*, $\sum_{\mu=\ell}^{q} \tilde{P}_\mu^{LPT}$ equals the total amount of work after u_ℓ in S^{LPT}, while $\sum_{\mu=\ell}^{q} P^*_\mu$ is the same in the optimal schedule S^*.

Proof (of Claim 3) First we prove the claim for each ℓ such that $\tilde{P}_\ell^{LPT} \neq 0$. Consider such an ℓ. If $\sum_{\mu=\ell}^{q} P^*_\mu$ were less than $p_{j(\ell)}$, then each job with a processing time at least $p_{j(\ell)}$ would be scheduled before u_ℓ in S^*, thus $\sum_{\nu=1}^{\ell-1} b'_\nu$ would be at least the total processing time of these jobs. However, this would mean that $j(\ell)$ could be scheduled earlier (recall that the machine is idle just before u_ℓ in S^{LPT}), thus we have $\sum_{\mu=\ell}^{q} P^*_\mu \ge p_{j(\ell)}$. Since $\tilde{P}_\ell^{LPT} \neq 0$, we can use Claim 2 (i) and we have

$$\sum_{\mu=\ell}^{q} \tilde{P}_\mu^{LPT} \le \sum_{\mu=\ell}^{q} P^*_\mu + p_{j(\ell)} \le 2 \cdot \sum_{\mu=\ell}^{q} P^*_\mu$$

Now suppose that $\tilde{P}_\ell^{LPT} = 0$. If $\sum_{\mu=\ell}^{q} \tilde{P}_\mu^{LPT} = 0$, then the claim is trivial. Otherwise, let $\ell' > \ell$ be the smallest index such that $\tilde{P}_{\ell'}^{LPT} \neq 0$. Since we know that the claim is true for ℓ', we have

$$\sum_{\mu=\ell}^{q} \tilde{P}_\mu^{LPT} = \sum_{\mu=\ell'}^{q} \tilde{P}_\mu^{LPT} \le 2 \cdot \sum_{\mu=\ell'}^{q} P^*_\mu \le 2 \cdot \sum_{\mu=\ell}^{q} P^*_\mu$$

and we are ready. □

Finally, as we have already noted, the LPT ordering of the jobs yields at least as good a schedule for I as the same job order for I', and the theorem is proved. □

Tight example. For any integer $n \ge 3$ consider the scheduling problem with n jobs, the first $n-1$ jobs are of unit processing time, while the last job has processing time n. That is, $p_j = a_j = w_j = 1$ for $j = 1, \ldots, n-1$, and $p_n = a_n = w_n = n$ for job n. There are two supplies, one at $u_1 = 0$ with supplied quantity $n-1$, and another at $u_2 = n^2$ with supplied quantity n. In the optimal schedule, the first $n-1$ jobs are scheduled from time 0, and the last job is scheduled at time n^2 (at u_2). That is, $C_j^* = j$ for $j = 1, \ldots, n-1$, and $C_n^* = n^2 + n$. The optimal objective function value is

$$\sum_{j=1}^{n} p_j C_j^* = n(n-1)/2 + (n^3 + n^2).$$

In contrast, in the LPT schedule job n comes first, but it can be scheduled only at time $u_2 = n^2$, since its demand is n. Hence, $C_n^{LPT} = n^2 + n$, and $C_j^{LPT} = n^2 + n + j$ for $j = 1, \ldots, n-1$. Consequently,

$$\begin{aligned}\sum_{j=1}^{n} p_j C_j^{LPT} &= (n^3 + n^2) + \sum_{j=1}^{n-1} (n^2 + n + j) \\ &= (n^3 + n^2) + (n^2 + n)(n-1) + n(n-1)/2.\end{aligned}$$

Therefore, the relative error of LPT on these instances is

$$\frac{(n^3 + n^2) + (n^2 + n)(n-1) + n(n-1)/2}{n(n-1)/2 + (n^3 + n^2)} = \frac{2n^3 + O(n^2)}{n^3 + O(n^2)},$$

which tends to 2 as n goes to infinity.

6 PTAS for $1|nr = 1, p_j = w_j, q = const| \sum w_j C_j$

Now we consider the special case when the number of supply dates is a constant (not part of the input), and at least 3 (for $q = 2$, there is an FPTAS for the general problem $1|nr = 1, q = 2| \sum w_j C_j$ [15]), and $p_j = w_j$ for each job j. Theorem 3 implies that this version is still NP-hard. However, below we describe a PTAS for it.

Let $P_{\text{sum}} := \sum_j p_j$ be the total processing time of the jobs. Let $\Delta := 1 + (\varepsilon/q^2)$. We will guess the total processing time of those jobs scheduled after u_ℓ for $\ell = 2, \ldots, q$, where a guess is a $q-1$ dimensional vector of non-increasing numbers $P_2^g, \ldots, P_q^g$, i.e., $P_\ell^g \geq P_{\ell+1}^g \geq 1$ for $\ell = 2, \ldots, q-1$, and each P_ℓ^g is of the form Δ^t for some integer $t \geq 0$ with $\Delta^t \leq P_{\text{sum}}$. Also fix $P_1^g := P_{\text{sum}}$. For any guess, define the set of *medium size jobs* $\mathcal{M}_\ell := \{j \mid p_j \geq (\Delta-1)P_\ell^g\}$. Note that $\mathcal{M}_q \supseteq \mathcal{M}_{q-1} \supseteq \cdots \supseteq \mathcal{M}_1$, since $P_q^g \leq P_{q-1}^g \leq \cdots \leq P_1^g$. Let $\mathcal{S}_\ell$ be the complement of $\mathcal{M}_\ell$, i.e., $\mathcal{S}_\ell := \{j \mid p_j < (\Delta - 1)P_\ell^g\}$. Clearly, $\mathcal{S}_q \subseteq \mathcal{S}_{q-1} \subseteq \cdots \subseteq \mathcal{S}_1$. After these preliminaries, the PTAS for $1|nr = 1, p_j = w_j, q = const| \sum w_j C_j$ consists of the following steps:

1. Consider each possible guess $(P_2^g, \ldots, P_q^g)$ of the total processing time of those jobs starting after the supply dates $u_2, \ldots, u_q$, respectively. For each possible guess define the sets of jobs $\mathcal{M}_\ell$ and $\mathcal{S}_\ell$ (see above), and perform the steps 2-5. After processing all the guesses, go to Step 6.
2. For each $\ell = 1, \ldots, q$, choose at most $1/(\Delta - 1)$ medium size jobs from $\mathcal{M}_\ell$ (since the sets $\mathcal{M}_\ell$ are not disjoint, care must be taken to choose each job at most once). For each possible choice $(T_1, \ldots, T_q)$ of the medium size jobs (where $T_\ell \subseteq \mathcal{M}_\ell$), perform steps 3-5. After evaluating all choices, continue with the next guess in Step 1.
3. Determine a schedule of the medium jobs. That is, for $\ell = 1, \ldots, q$, schedule the jobs in T_ℓ in any order contiguously after u_ℓ, and after all the previously scheduled jobs.
4. Let $\mathcal{J}_0^u$ be the set of unscheduled jobs. For $\ell = q, q-1, \ldots, 1$, repeat the following. In a general step with $\ell \geq 2$, pick jobs from $\mathcal{J}_{q-\ell}^u \cap \mathcal{S}_\ell$ in non-increasing a_j/p_j order until the selected subset K_ℓ satisfies $p(K_\ell)+p(T_\ell) \geq P_\ell^g - (1/\Delta)P_{\ell+1}^g$, or no more jobs are left, i.e., $K_\ell = \mathcal{J}_{q-\ell}^u \cap \mathcal{S}_\ell$. In either case, insert the jobs of K_ℓ in any order after u_ℓ and after all the jobs in $T_1 \cup \cdots \cup T_{\ell-1}$, and before all the jobs in $T_\ell \cup \bigcup_{\ell'=\ell+1}^q (K_{\ell'} \cup T_{\ell'})$ (pushing some of them to the right if necessary). Let $\mathcal{J}_{q-\ell+1}^u := \mathcal{J}_{q-\ell}^u \setminus K_\ell$ and continue with $\ell - 1$ until $\ell = 1$ or no more unscheduled jobs are left. For $\ell = 1$ just schedule all the remaining jobs from time $u_1 = 0$ on (pushing the already scheduled jobs to the right, if necessary). If the complete schedule obtained satisfies the resource constraints, then continue with Step 5, otherwise with the next choice of medium size jobs in Step 2.
5. Compute the objective function value of the complete schedule obtained in step (4), and store this schedule as the best schedule if it is the first feasible schedule or if it is better than the best feasible schedule found so far. Continue with next choice of medium size jobs in Step 2.
6. Output the best schedule found in the previous steps.

Theorem 5 *The above algorithm is a PTAS for* $1|nr = 1, p_j = w_j, q = const| \sum w_j C_j$.

Proof Let I be any instance of the scheduling problem, and S^* an optimal solution for I. Let $\hat{P}_\ell^*$ be the total processing time of those jobs starting after u_ℓ in S^*. Clearly, $\hat{P}_\ell^* \geq \hat{P}_{\ell+1}^*$ for $\ell = 1, \ldots, q-1$. Consider the guess $P_2^g, \ldots, P_q^g$ in Step 1 of our algorithm such that $\hat{P}_\ell^* \leq P_\ell^g < \Delta \hat{P}_\ell^*$ for each $\ell = 2, \ldots, q$. Such a guess must exist by the definition of guesses.

For each $\ell = 1, \ldots, q$, let us partition the set of jobs that start in the interval $[u_\ell, u_\ell + 1)$ in the schedule S^* into subsets $T_\ell^* \subseteq \mathcal{M}_\ell$ and $K_\ell^* \subseteq \mathcal{S}_\ell$. Clearly, the sets T_ℓ^* are disjoint, and the cardinality of each T_ℓ^* is at most $1/(\Delta - 1)$, since $P_\ell^g \geq P_\ell^*$ and thus each job in T_ℓ^* is of size at least $(\Delta-1)P_\ell^g \geq (\Delta-1)\hat{P}_\ell^*$, while the total size of all the jobs starting after u_ℓ in S^* is $\hat{P}_\ell^*$ by definition. Therefore, the algorithm will enumerate and process the choice $(T_1^*, \ldots, T_q^*)$ in Step 2. In the rest of the proof we fix this choice of medium jobs. After scheduling them in Step 3, the resulting schedule is like S^*, except that some jobs may be unscheduled yet. Thus we perform Step 4, and let S^A be the resulting schedule. In Step 4 the algorithm will find sets of jobs $K_1, \ldots, K_q$, and it may well be the case that $K_\ell^* \neq K_\ell$ for some ℓ, but we know that $\bigcup_{\ell=1}^q K_\ell = \bigcup_{\ell=1}^q K_\ell^*$, since in S^* and S^A, for each ℓ, the same subset T_ℓ^* of $\mathcal{M}_\ell$ is chosen. We will prove that S^A is a feasible schedule and that its objective function value is at most $1 + O(\varepsilon)$ times the optimum.

Claim 4 *The total processing time of those jobs that start after* u_ℓ *in* S^*, *and in* S^A, *respectively, satisfy the inequalities*

$$\hat{P}_\ell^* \leq \sum_{\ell'=\ell}^{q} \sum_{j \in T_{\ell'}^* \cup K_{\ell'}} p_j \leq (1 + 6(\varepsilon/q))\hat{P}_\ell^*, \quad \ell = 2, \ldots, q. \tag{3}$$

Proof First notice that for each $\ell = 2, \ldots, q$, we have $\cup_{\ell'=\ell}^{q} K_{\ell'}^* \subseteq \mathcal{S}_\ell \setminus \left(\bigcup_{\ell'=\ell+1}^{q} T_{\ell'}^*\right)$, since $K_{\ell'}^* \subseteq \mathcal{S}_{\ell'} \subseteq \mathcal{S}_\ell$ and $T_{\ell'}^* \cap K_{\ell'}^* = \emptyset$ for $\ell' \geq \ell$, and similarly, $\cup_{\ell'=\ell}^{q} K_{\ell'} \subseteq \mathcal{S}_\ell \setminus \left(\bigcup_{\ell'=\ell+1}^{q} T_{\ell'}^*\right)$. We prove (3) by induction. Along with (3), we will also prove

$$p(T_\ell^*) + p(K_\ell) \leq \Delta P_\ell^g - (1/\Delta) P_{\ell+1}^g, \quad \ell = 2, \ldots, q, \tag{4}$$

where we define $P_{q+1}^g := 0$. The base case is for $\ell = q+1$, when all the inequalties trivially hold (we define $\hat{P}_{q+1}^* := 0$). Now suppose that (3) and (4) hold for $\ell+1$ for some $\ell \geq 2$, and we verfy them for ℓ.

First suppose that $p(K_\ell) + p(T_\ell^*) \geq P_\ell^g - (1/\Delta) P_{\ell+1}^g$. Since $\hat{P}_\ell^* - \hat{P}_{\ell+1}^*$ equals the total processing time of those jobs that start in the interval $[u_\ell, u_{\ell+1})$ in S^*, and $\hat{P}_\ell^* \leq P_\ell^g < \Delta \hat{P}_\ell^*$, we have

$$\begin{aligned} p(T_\ell^*) + p(K_\ell^*) = \hat{P}_\ell^* - \hat{P}_{\ell+1}^* &\leq P_\ell^g - (1/\Delta) P_{\ell+1}^g \\ &\leq p(T_\ell^*) + p(K_\ell). \end{aligned} \tag{5}$$

So, the induction hypothesis implies the first inequality in (3). To verify the second one, recall that in Step 4 we stop selecting jobs as soon as $p(T_\ell^*) + p(K_\ell)$ exceeds $P_\ell^g - (1/\Delta) P_{\ell+1}^g$, and the processing time of all jobs in $\mathcal{S}_\ell$ is bounded by $(\Delta - 1) P_\ell^g$, which implies (4) for ℓ, since

$$\begin{aligned} p(T_\ell^*) + p(K_\ell) &< P_\ell^g - (1/\Delta) P_{\ell+1}^g + (\Delta - 1) P_\ell^g \\ &= \Delta P_\ell^g - (1/\Delta) P_{\ell+1}^g. \end{aligned}$$

Using the induction hypothesis, we obtain

$$\begin{aligned} \sum_{\ell'=\ell}^{q} (p(T_{\ell'}^*) + p(K_{\ell'})) &\leq \sum_{\ell'=\ell}^{q} (\Delta P_{\ell'}^g - (1/\Delta) P_{\ell'+1}^g) \\ &< \Delta^2 P_\ell^g + \sum_{\ell'=\ell+1}^{q} (\Delta^2 - 1) P_{\ell'}^g. \end{aligned} \tag{6}$$

A simple calculation shows that $\Delta^2 < 1 + 3(\varepsilon/q^2) < 1 + (\varepsilon/q)$ (since $q \geq 3$ by assumption), therefore, the right-hand-side of (6) is less than $(1 + (\varepsilon/q)) P_\ell^g + 3(\varepsilon/q^2) \sum_{\ell'=\ell+1}^{q} P_{\ell'}^g \leq (1 + 4(\varepsilon/q)) P_\ell^g$. Since $P_\ell^g < \Delta \hat{P}_\ell^*$, and $(1 + 4(\varepsilon/q))\Delta < 1 + 6(\varepsilon/q)$ (since $q \geq 3$ by assumption), the second inequality in (3) follows.

Now suppose $K_\ell = \mathcal{J}_{q-\ell}^u \cap \mathcal{S}_\ell$ and $p(K_\ell) + p(T_\ell^*) < P_\ell^g - (1/\Delta) P_{\ell+1}^g$ in Step 4 of the algorithm at iteration ℓ. Then we deduce that in S^A, all the small jobs in $\mathcal{S}_\ell \setminus \left(\bigcup_{\ell'=\ell+1}^{q} T_{\ell'}^*\right)$ are scheduled after u_ℓ in the iterations $\ell, \ldots, q$, while in S^*, some jobs of $\mathcal{S}_\ell \setminus \left(\bigcup_{\ell'=\ell+1}^{q} T_{\ell'}^*\right)$ may be started before u_ℓ. Therefore, the first inequality in (3) holds in this case as well. To verify the second inequality in (3), note that since $p(K_\ell) + p(T_\ell^*) < P_\ell^g - (1/\Delta) P_{\ell+1}^g$ by assumption, (4) follows immediately. Then, using the induction hypothesis, we obtain (6), and then the same argument applies as above. □

In order to prove (resource) feasibility, we need some further technical results. To simplify notation, suppose $\mathcal{S}_1 \setminus \bigcup_{\ell=2}^{q} T_\ell^* = \{1, \ldots, n_1\}$ and $a_j/p_j \geq a_{j+1}/p_{j+1}$ for $1 \leq j < n_1$, i.e., job j is the j^{th} job in the ordered sequence. Let $X_t := \{1, \ldots, t\}$ be the index set of the first $t \leq n_1$ jobs with the largest a_j/p_j ratio.

Claim 5 *There exists a unique $t \in \{0, \ldots, n_1\}$ such that*

$$\bigcup_{\ell'=\ell}^{q} K_{\ell'} = X_t \cap \left(\mathcal{S}_\ell \setminus \bigcup_{\ell'=\ell}^{q} T_{\ell'}^*\right) \tag{7}$$

Proof If $\bigcup_{\ell'=\ell}^{q} K_{\ell'}$ is the empty set, then $t = 0$ will do. Otherwise, let t be the maximum element in $\bigcup_{\ell'=\ell}^{q} K_{\ell'}$. Indirectly, suppose there exists some $t' < t$ such that $t' \notin \bigcup_{\ell'=\ell}^{q} K_{\ell'}$, but $t' \in \mathcal{S}_\ell \setminus \bigcup_{\ell'=\ell}^{q} T_{\ell'}^*$. Then, at some iteration in Step 4 of the algorithm, t would be chosen in place of $t' < t$, which is a contradiction. □

Corollary 1 *For the job index t defined in Claim 5,*

$$\left(\bigcup_{\ell'=\ell}^{q} K_{\ell'}^*\right) \cap \left(\bigcup_{\ell'=\ell}^{q} K_{\ell'}\right) = X_t \cap \left(\bigcup_{\ell'=\ell}^{q} K_{\ell'}^*\right).$$

Claim 6 *For each $t = 1, \ldots, n_1$, and $2 \leq \ell \leq q$, we have*

$$\sum_{j \in X_t \cap (\cup_{\ell'=\ell}^{q} K_{\ell'})} p_j \geq \sum_{j \in X_t \cap (\cup_{\ell'=\ell}^{q} K_{\ell'}^*)} p_j \tag{8}$$

Proof We proceed by induction, the base case being for $\ell = q$. Then $K_q, K_q^* \subseteq \mathcal{S}_q$. If K_q is a proper subset of $\mathcal{S}_q$, then we have $p(K_q^*) \leq p(K_q)$ by (5). Otherwise $K_q = \mathcal{S}_q \supseteq K_q^*$, and we have $p(K_q^*) \leq p(K_q)$ in this case, too. For the sake of a contradiction, suppose there exists $1 \leq t \leq n_1$ such that (8) does not hold. Let t be the smallest such job index. Then job $t \in K_q^* \setminus K_q$, otherwise t could be decreased. Then K_q does not contain any job v with $v > t$, otherwise, before picking v, the algorithm would have picked t. But then

$$\sum_{j \in K_q \cap X_t} p_j = \sum_{j \in K_q} p_j \geq \sum_{j \in K_q^*} p_j \geq \sum_{j \in K_q^* \cap X_t} p_j,$$

which is a contradiction.

Now assume by induction that (8) holds for $\ell = k+1$, with $k \geq 2$, and for all $1 \leq t \leq n_1$, and we check it for $\ell = k$. We distinguish two cases.

- $K_\ell \subset \mathcal{S}_\ell \setminus \bigcup_{\ell'=\ell+1}^{q} (T_{\ell'}^* \cup K_{\ell'})$. Then we have $p(K_\ell^*) \leq p(K_\ell)$ by (5). For the sake of a contradiction, suppose there exists $1 \leq t \leq n_1$ such that (8) does not hold. Let t be the smallest such job index. Then it must be the case that $t \in (K_\ell^* \cup \cdots \cup K_q^*) \setminus$

$(K_\ell \cup \cdots \cup K_q)$, otherwise t could be decreased. So suppose $t \in K^*_{\ell'}$ for some $\ell \le \ell' \le q$. Then $\{t, \dots, n_1\} \cap K_\ell = \emptyset$, because if not, then, since $t \in K^*_{\ell'} \subseteq \mathcal{S}_{\ell'} \subseteq \mathcal{S}_\ell$, the algorithm would have chosen t before picking some $v \in \{t+1, \dots, n_1\} \cap K_\ell$. Consequently, $K_\ell \subseteq X_{t-1}$. Now we use the induction hypothesis:

$$\begin{aligned}\sum_{j \in X_t \cap (K_\ell \cup \cdots \cup K_q)} p_j &= \sum_{j \in K_\ell} p_j + \sum_{j \in X_t \cap (K_{\ell+1} \cup \cdots \cup K_q)} p_j \\ &\ge \sum_{j \in K_\ell} p_j + \sum_{j \in X_t \cap (K^*_{\ell+1} \cup \cdots \cup K^*_q)} p_j \\ &\ge \sum_{j \in K^*_\ell \cap X_t} p_j + \sum_{j \in X_t \cap (K^*_{\ell+1} \cup \cdots \cup K^*_q)} p_j,\end{aligned}$$

where the first equation follows from $K_\ell \subseteq X_{t-1} \subset X_t$, the first inequality from the induction hypothesis, and the last inequality from the fact that $p(K_\ell) \ge p(K^*_\ell)$. However, the derived inequality is just (8) for ℓ and t, a contradiction.

– $K_\ell = \mathcal{S}_\ell \setminus \bigcup_{\ell'=\ell+1}^q (T^*_{\ell'} \cup K_{\ell'})$. Since $\bigcup_{\ell'=\ell}^q K^*_{\ell'} \subseteq \mathcal{S}_\ell \setminus \bigcup_{\ell'=\ell+1}^q T^*_{\ell'}$, we can observe that each $t \in \mathcal{S}_\ell \setminus \bigcup_{\ell'=\ell+1}^q T^*_{\ell'}$ belongs to one of the sets $K_{\ell'}$ with $\ell \le \ell' \le q$, but may not belong to any of the sets $K^*_{\ell'}$ with $\ell \le \ell' \le q$. Hence, the claim follows in this case, too.

□

Corollary 2 *For each $\ell = 2, \dots, q$, we have*

$$\sum_{j \in \bigcup_{\ell'=\ell}^q K_{\ell'}} p_j \ge \sum_{j \in \bigcup_{\ell'=\ell}^q K^*_{\ell'}} p_j.$$

Now we verify resource feasibility by showing that for each $\ell = 2, \dots, q$,

$$\sum_{j \in \bigcup_{\ell'=\ell}^q K_{\ell'}} a_j \ge \sum_{j \in \bigcup_{\ell'=\ell}^q K^*_{\ell'}} a_j. \tag{9}$$

This suffices to prove the feasibility of S^A, because then for each $\ell = 2, \dots, q$, the total resource consumption of those jobs that start after u_ℓ in S^A is at least as much as that in S^*. Therefore, the total resource consumption of those jobs that start not later than u_ℓ in S^A cannot be more than that in S^*. Hence, S^A is a feasible schedule. Let t be the job index defined in Claim 5. Now we compute

$$\begin{aligned}\sum_{j \in \bigcup_{\ell'=\ell}^q K^*_{\ell'}} a_j &\underset{(a)}{=} \sum_{j \in X_t \cap \left(\bigcup_{\ell'=\ell}^q K^*_{\ell'}\right)} a_j + \sum_{j \in \overline{X}_t \cap \left(\bigcup_{\ell'=\ell}^q K^*_{\ell'}\right)} \left(\frac{a_j}{p_j}\right) \cdot a_j \\ &\underset{(b)}{\le} \sum_{j \in X_t \cap \left(\bigcup_{\ell'=\ell}^q K^*_{\ell'}\right)} a_j + \max_{j \in \overline{X}_t \cap \left(\bigcup_{\ell'=\ell}^q K^*_{\ell'}\right)} \frac{a_j}{p_j} \left(\sum_{j \in \overline{X}_t \cap \left(\bigcup_{\ell'=\ell}^q K^*_{\ell'}\right)} p_j \right) \\ &\underset{(c)}{\le} \sum_{j \in \left(\bigcup_{\ell'=\ell}^q K^*_{\ell'}\right) \cap \left(\bigcup_{\ell'=\ell}^q K_{\ell'}\right)} a_j + \min_{j \in \left(\bigcup_{\ell'=\ell}^q K_{\ell'}\right) \setminus \left(\bigcup_{\ell'=\ell}^q K^*_{\ell'}\right)} \frac{a_j}{p_j} \left(\sum_{j \in \left(\bigcup_{\ell'=\ell}^q K_{\ell'}\right) \setminus \left(\bigcup_{\ell'=\ell}^q K^*_{\ell'}\right)} p_j \right) \\ &\underset{(d)}{\le} \sum_{j \in \bigcup_{\ell'=\ell}^q K_{\ell'}} a_j,\end{aligned}$$

where (a), (b) and (d) are obvious, and (c) follows from three observations:

(i) the first terms of the two expressions are the same by Corollary 1,

(ii) the inequality between the second terms follows from

$$\max_{j \in \overline{X}_t \cap \left(\bigcup_{\ell'=\ell}^q K^*_{\ell'}\right)} \frac{a_j}{p_j} \le \min_{j \in \left(\bigcup_{\ell'=\ell}^q K_{\ell'}\right) \setminus \left(\bigcup_{\ell'=\ell}^q K^*_{\ell'}\right)} \frac{a_j}{p_j},$$

since the jobs are indexed in non-increasing a_j/p_j order, and $\bigcup_{\ell'=\ell}^q K_{\ell'} \subseteq X_t$, and from

(iii)

$$\sum_{j \in \overline{X}_t \cap \left(\bigcup_{\ell'=\ell}^q K^*_{\ell'}\right)} p_j \le \sum_{j \in \left(\bigcup_{\ell'=\ell}^q K_{\ell'}\right) \setminus \left(\bigcup_{\ell'=\ell}^q K^*_{\ell'}\right)} p_j,$$

by Corollaries 1 and 2.

Now we bound the objective function value of S^A. Again, we need a technical result. Let H^A_ℓ denote the idle time in $[u_\ell, u_{\ell+1})$ in the schedule S^A, and G^A_ℓ the total idle time before $u_{\ell+1}$.

Claim 7 $H^A_\ell \le H^*_\ell + (6\varepsilon/q)\hat{P}^*_{\ell+1}$.

Proof Observe that in S^A at most $(6\varepsilon/q)\hat{P}^*_{\ell+1}$ more work is scheduled after $u_{\ell+1}$ than in S^* by inequality (3). Therefore, the total gap in S^A before $u_{\ell+1}$ is at most $(6\varepsilon/q)\hat{P}^*_{\ell+1}$ more than in S^*, i.e.,

$$G^A_\ell \le G^*_\ell + (6\varepsilon/q)\hat{P}^*_{\ell+1}.$$

On the other hand, $G^*_{\ell-1} \le G^A_{\ell-1}$, since in S^A, $\sum_{\ell'=\ell}^{q}(p(T^*_{\ell'}) + p(K_{\ell'})) \ge \hat{P}^*_\ell$ by (3). Now, using the fact that $G^A_\ell = G^A_{\ell-1} + H^A_\ell$, we obtain

$$\begin{aligned} H^A_\ell &= G^A_\ell - G^A_{\ell-1} \le G^*_\ell + (6\varepsilon/q)\hat{P}^*_\ell - G^*_{\ell-1} \\ &= H^*_\ell + (6\varepsilon/q)\hat{P}^*_{\ell+1}. \end{aligned}$$

□

Now we compute:

$$\begin{aligned} &\sum_{j\in\mathcal{J}} p_j C^A_j = \sum_{j\le k} p_j p_k + \sum_{\ell=2}^{q} H^A_{\ell-1} \cdot \sum_{\ell'=\ell}^{q} p(T^*_{\ell'} \cup K_{\ell'}) \\ &\le \sum_{j\le k} p_j p_k + \sum_{\ell=2}^{q} (H^*_{\ell-1} + (6\varepsilon/q)\hat{P}^*_\ell)(1 + (6\varepsilon/q))\hat{P}^*_\ell \\ &\le \sum_{j\le k} p_j p_k + (1 + (6\varepsilon/q)) \sum_{\ell=2}^{q} H^*_{\ell-1}\hat{P}^*_\ell + O(\varepsilon)(P_{\mathrm{sum}})^2 \\ &\le \sum_{j\le k} p_j p_k + (1 + (6\varepsilon/q)) \sum_{\ell=2}^{q} H^*_{\ell-1}\hat{P}^*_\ell + O(\varepsilon) \sum_{j\le k} p_j p_k \\ &\le (1 + O(\varepsilon)) \sum_{j\in\mathcal{J}} p_j C^*_j. \end{aligned}$$

It remains to verify the running time of the algorithm. The number of guesses in Step 1 is $O((\log_\Delta P_{\mathrm{sum}})^q)$ which is bounded by $O(((q^2/\varepsilon)\ln P_{\mathrm{sum}})^q)$, which is polynomial in the size of the input. The number of choices in Step 2 is bounded by $O(n^{q^3/\varepsilon})$. The rest can be done in $O(n^2)$ time for every guess $(P^g_2, \ldots, P^g_q)$ and choice of jobs $(T_1, \ldots, T_q)$. Hence, the total time complexity is polynomial bounded in the size of the input. □

7 Final remarks

In this paper we have established new complexity, and approximability results for single machine scheduling problems with non-renewable resource constraints and the total weighted completion time objective. As it has turned out, List Scheduling is a useful tool in solving a number of special cases, and it can also be the basis of designing approximation algorithms.

There are a number of open problems. For instance, is there a polynomial time approximation algorithm of constant approximation ratio for the problem $1|nr = 1|\sum w_j C_j$? And for the special case $1|nr = 1, p_j = 1, w_j = a_j|\sum w_j C_j$?

Acknowledgments

This work has been supported by the National Research, Development and Innovation Office – NKFIH, grant no. K112881, and by the GINOP-2.3.2-15-2016-00002 grant of the Ministry of National Economy of Hungary.

References

1. Jacek Blazewicz, Klaus H Ecker, Erwin Pesch, Günter Schmidt, and Jan Weglarz. *Scheduling computer and manufacturing processes*. Springer Science & Business media, 2013.
2. D. Briskorn, B.-C. Choi, K. Lee, J. Leung, and M. Pinedo. Complexity of single machine scheduling subject to nonnegative inventory constraints. *European Journal of Operational Research*, 207:605–619, 2010.
3. D. Briskorn, F. Jaehn, and E. Pesch. Exact algorithms for inventory constrained scheduling on a single machine. *Journal of Scheduling*, 16:105–115, 2013.
4. Jacques Carlier. *Problèmes d'ordonnancements à contraintes de ressources: algorithmes et complexité. Thèse d'état*. Université Paris 6, 1984.
5. 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*, 62:7–13, 2011.
6. Ronald L Graham, Eugene L Lawler, Jan Karel Lenstra, and AHG Rinnooy Kan. Optimization and approximation in deterministic sequencing and scheduling: a survey. *Annals of discrete mathematics*, 5:287–326, 1979.
7. A. Grigoriev, M. Holthuijsen, and J. van de Klundert. Basic scheduling problems with raw material constraints. *Naval Research of Logistics*, 52:527–553, 2005.
8. Péter Györgyi. A ptas for a resource scheduling problem with arbitrary number of parallel machines. *Operations Research Letters*, 45:604–609, 2017.
9. Péter Györgyi and Tamás Kis. Approximation schemes for single machine scheduling with non-renewable resource constraints. *Journal of Scheduling*, 17:135–144, 2014.
10. Péter Györgyi and Tamás Kis. Approximability of scheduling problems with resource consuming jobs. *Annals of Operations Research*, 235(1):319–336, 2015.
11. Péter Györgyi and Tamás Kis. Reductions between scheduling problems with non-renewable resources and knapsack problems. *Theoretical Computer Science*, 565:63–76, 2015.
12. Péter Györgyi and Tamás Kis. Approximation schemes for parallel machine scheduling with non-renewable resources. *European Journal of Operational Research*, 258(1):113 – 123, 2017.
13. Péter Györgyi and Tamás Kis. Minimizing the maximum lateness on a single machine with raw material constraints by branch-and-cut. *Computers & Industrial Engineering*, 115:220–225, 2018.
14. Hans Kellerer, Vladimir Kotov, Franz Rendl, and Gerhard J Woeginger. The stock size problem. *Operations Research*, 46(3):S1–S12, 1998.
15. Tamás Kis. Approximability of total weighted completion time with resource consuming jobs. *Operations Research Letters*, 43(6):595–598, 2015.

16. Ehab Morsy and Erwin Pesch. Approximation algorithms for inventory constrained scheduling on a single machine. *Journal of Scheduling*, 18(6):645–653, 2015.
17. Roman Slowinski. Preemptive scheduling of independent jobs on parallel machines subject to financial constraints. *European Journal of Operational Research*, 15:366–373, 1984.
18. Wayne E Smith. Various optimizers for single-stage production. *Naval Research Logistics (NRL)*, 3(1-2):59–66, 1956.
19. A. Toker, S. Kondakci, and N. Erkip. Scheduling under a non-renewable resource constraint. *Journal of the Operational Research Society*, 42:811–814, 1991.
20. Jinxing Xie. Polynomial algorithms for single machine scheduling problems with financial constraints. *Operations Research Letters*, 21(1):39–42, 1997.