RAIRO-Oper. Res. 52 (2018) 315–334
https://doi.org/10.1051/ro/2017040

RAIRO Operations Research
www.rairo-ro.org

SINGLE-MACHINE BATCH SCHEDULING PROBLEM WITH JOB REJECTION AND RESOURCE DEPENDENT PROCESSING TIMES

Weifan Huang[1], Chin-Chia Wu[2] and Shangchia Liu[3,*]

Abstract. This paper addresses single-machine batch scheduling with job rejection and convex resource allocation. A job is either rejected, in which case a rejection penalty will be incurred, or accepted and processed on the machine. The accepted jobs are combined to form batches containing contiguously scheduled jobs. For each batch, a batch-dependent machine setup time, which is a function of the number of batches processed previously, is required before the first job in the batch is processed. Both the setup times and job processing times are controllable by allocating a continuously divisible nonrenewable resource to the jobs. The objective is to determine an accepted job sequence, a rejected job set, a partition of the accepted job sequence into batches, and resource allocation that jointly minimize a cost function based on the total delivery dates of the accepted jobs, and the job holding, resource consumption, and rejection penalties. An dynamic programming solution algorithm with running time $O(n^6)$ is developed for the problem. It is also shown that the case of the problem with a common setup time can be solved in $O(n^5)$ time.

Mathematics Subject Classification. 90B35, 90C26.

Received October 18, 2016. Accepted May 17, 2017.

1. Introduction

1.1. Problem definition

The problem of single-machine batch scheduling with job rejection and convex resource allocation can be stated as follows: There is a set of n independent jobs $N = \{J_1, J_2, \ldots, J_n\}$ to be processed on a single machine. The machine can handle at most one job at a time and job preemption is not allowed. Each job J_j becomes available for processing at time zero, requires a nonnegative processing time p_j and has a nonnegative rejection penalty e_j. For each job one must decide whether to accept and schedule it on the machine or reject it. If job J_j is rejected, there will incur a rejection penalty e_j. Denote the sets of accepted jobs and rejected jobs by A and $\overline{A}$, respectively. The accepted jobs in A are combined to form batches containing contiguously scheduled jobs. The jobs assigned to the same batch are processed contiguously and are delivered to customers together when the processing of the last job in the batch is finished. Thus, the delivery date of each accepted job is equal to its assigned batch delivery date. When a job is finished before its delivery date, it must wait in the

Keywords. Scheduling, batching, resource allocation, rejected penalty.

[1] School of Mathematics and Computer Science, Yichun University, Yichun, Jiangxi 336000, P.R. China.
[2] Department of Statistics, Feng Chia University, Taichung, Taiwan.
[3] Department of Business Administration, Fu Jen Catholic University,Hsinpei City, Taiwan.
* Corresponding author:docchia@gmail.com

Article published by EDP Sciences

system until delivery. Thus, an inventory holding cost is incurred for each early job and this cost is assumed to be proportional to the earliness value of the job with respect to its batch delivery date. For each batch, a batch-dependent machine setup time, which is a function of the number of batches processed previously, is required before the first job in the batch is processed, *i.e.*, the setup of the ith batch depends solely on its index i, denoted by s_i. Since there is no need to schedule the set of rejected jobs, a schedule can be represented by a job sequence (permutation) $\pi = (S, \overline{A})$ in which the first $n_a = |A|$ jobs correspond to set A in the order of S with the corresponding partition of this sequence into batches and the last $n - n_a = |\overline{A}|$ jobs correspond to set $\overline{A}$ in any arbitrary sequence.

The setup time s_i may be compressed if an amount u_i of a continuous and nonrenewable resource is used to perform the setup i:

$$s_i(u_i) = \left(\frac{\omega_i}{u_i}\right)^k, \tag{1.1}$$

where ω_i is a positive parameter, which represents the workload of the setup operation for the ith batch, u_i is the amount of the resource allocated to perform setup i, and k is a positive constant. Similarly, the processing time p_j may also be compressed if an amount v_j of a continuous and nonrenewable resource is used to perform job J_j:

$$p_j(v_j) = \left(\frac{w_j}{v_j}\right)^k, \tag{1.2}$$

where w_j is a positive parameter, which represents the workload of the processing operation for job J_j, v_j is the amount of the resource allocated to process job J_j, and k is a positive constant. It assumes that the technologies deployed to reduce the setup and processing times are closely related in the sense that the functions controlling their behaviours share the same exponent k.

The problem in this paper is to determine an accepted job sequence $S = (J_{[1]}, \ldots, J_{[n_a]})$, a rejected job set $\overline{A}$, a partition of the accepted job sequence S into batches $\mathbf{B} = (B_1, B_2, \ldots, B_m)$, and resource allocations $\mathbf{u} = (u_1, u_2, \ldots, u_m)$ and $\mathbf{v} = (v_{[1]}, v_{[2]}, \ldots, v_{[n_a]})$ such that the following objective function is minimized

$$Z(S, \overline{A}, \mathbf{B}, \mathbf{u}, \mathbf{v}) = \sum_{j=1}^{n_a} (\alpha D_{[j]} + \beta H_{[j]} + \delta_{[j]} v_{[j]}) + \sum_{i=1}^{m} \gamma_i u_i + \sum_{j \in \overline{A}} e_j, \tag{1.3}$$

where $\alpha \geq \beta$ and $[j]$ represents the job that is the jth job in S; for each accepted job J_j, C_j is the completion time, D_j is the delivery date ($D_j \geq C_j$), which is the completion time of the last job in its assigned batch, and $H_j = D_j - C_j$ is the holding time, which is the time that passes from the instant that the job finishes its processing to the instant that it is delivered; α and β are the unit costs of the delivery time and job holding time, respectively; γ_i and δ_j are the costs of allocating one unit of the resource to process setup i and job J_j, respectively. The condition $\alpha \geq \beta$ is reasonable because the delivery times affect customers more than holding finished jobs in stock. It is easy to see that $D_j = C_j$ or $D_j = C_l > C_j$, where job J_l is some job processed after J_j. Using the traditional notation for describing scheduling problems, denote the problem under study by $1|covx, rej, \mathbf{B}| \sum_{j=1}^{n_a} (\alpha D_{[j]} + \beta H_{[j]} + \delta_{[j]} v_{[j]}) + \sum_{i=1}^{m} \gamma_i u_i + \sum_{j \in \overline{A}} e_j$.

For a practical example, consider a situation in manufacturing that involves two parties: a manufacturer and a customer. A set of jobs (customer orders) must be processed in the manufacturer, referred to as a single "machine". The finished jobs are to be delivered to the customer in batches. Each batch is to be assigned a dispatch date on which all the jobs in the batch are to be delivered to the customer together. The jobs belonging to the same batch are executed serially and the dispatch date of a batch is equal to the completion time of the last job in the batch. A setup time, which may be batch-dependent, is needed to perform some cleaning operations

before starting a new batch. When a job is finished before its assigned batch dispatch date, it must wait in the system until delivery. Thus, an inventory holding cost is incurred for each early job. To improve customer service level usually expressed as a function of the times when the jobs are delivered, additional resources, such as money, energy, or manpower can allocated to control job processing times and setup times. In addition, because of production constraints at the manufacturer, the manufacturer may not be able to process all jobs. The manufacturer may prefer to process some jobs while reject some jobs, where each rejected job may either be outsourced or not get served at all at additional cost (rejection penalty) due to either the outsourcing cost or the loss in income. The problem is to find an optimal schedule of production, distribution and rejection with a composite objective that measures the customer service level and the total production cost. This situation can be modeled as our problem.

1.2. Relevant previous work

Batching problems, which combine the sequencing and partitioning problems, have attracted much research interest in recent years. The batching issues in scheduling have been discussed in Allahverdi *et al.* [1], Potts and Kovalyov [9], Potts and Van Wassenhove [10], Yin *et al.* [17,20]. The batching model studied in this paper is called the batch availability model in Potts and Kovalyov [9], by which all the jobs in a batch are processed contiguously and delivered to customers together when the processing of the last job in the batch is finished. This model arises in situations where jobs flow between processing facilities in containers such as pallets, boxes, or carts. A setup time is usually needed to remove a previous container, to install a new one, and to perform some cleaning operations. Potts and Kovalyov [9] discuss several generalizations of this basic case where the setup may be machine-dependent (in a multi-machine setting) and/or sequence-dependent. Indeed, in many real-life applications, the setup may change. In particular, the setup may be batch-dependent. Such applications include, *e.g.*, the very common case where the first setup time is larger than the following ones because it comprises several initialization procedures required for the entire production process. Some setups may be larger due to periodic maintenance procedures. In some cases, the setup times are arranged in nonincreasing order to reflect a learning effect. In other cases, the setup times are arranged in nondecreasing order to model machine deterioration. In order to reflect the above applications, Mosheiov and Oron [7] assume that the setup time is batch-dependent and consider the problem to minimize the total flowtime. They focus on the case of identical job processing times and provide an $O(n)$ solution procedure for the problem. In this paper it also assumed that the setup times are batch-dependent. However, the setup times can be compressed, and the job processing times are arbitrary nonnegative real and compressible by varying the allocation of some resources to the jobs.

In many real-life systems, the job processing times are controllable by allocating different amounts of resources, such as additional money, overtime, energy, fuel, catalysts, subcontracting, or additional manpower, to the jobs. In such systems, the job scheduling and resource allocation decisions should be carefully coordinated to achieve the most efficient system performance. The effect of resource allocation on the job processing time is usually described by a resource consumption function. Studies of scheduling problems with controllable processing times were initiated by Vickson [15,16]. Surveys of this area of scheduling research can be found in Nowicki and Zdrzalka [8] and Shabtay and Steiner [13]. In most studies of scheduling with controllable processing times, researchers assume that the job processing time is a bounded linear function of the amount of a resource allocated to process the job. For many resource allocation problems in physical or economic systems, however, the linear resource consumption function fails to reflect the law of diminishing marginal returns. This law states that productivity increases at a decreasing rate with the amount of resources employed. In order to model this, some studies of scheduling with resource allocation assume that the job processing time is a convex decreasing function of the amount of a resource allocated to process the job. This resource consumption function has been extensively used in continuous resource allocation theory. In this paper, the convex resource consumption function (see Eqs. (1.1) and (1.2)) is adopted. Recently, increasing studies focus on investigating the problems that simultaneously consider batch scheduling under batch availability and resource dependent processing times. For example, Cheng and Kovalyov [3] consider batch scheduling with deadlines and resource dependent processing times, where the processing time of a job is a linear decreasing function

of the amount of a common discrete resource allocated to it. The objective is to find simultaneously a resource allocation and a schedule which is feasible with respect to the deadlines so as to minimize the total weighted resource consumption. They show that the problem is NP-hard even for the special case of common parameters, and develop two dynamic programming algorithms for the general problem. Cheng *et al.* [4] investigate batch scheduling with deadlines and resource dependent processing times, where processing of a batch requires a machine setup time common for all batches, and both the job processing times and the setup time can be compressed through allocation of a continuously divisible resource. Polynomial time algorithms are presented to find an optimal batch sequence and resource values such that either the total weighted resource consumption is minimized, subject to meeting job deadlines, or the maximum job lateness is minimized, subject to an upper bound on the total weighted resource consumption. Shabtay and Steiner [14] study the single-machine batch scheduling problem, where both setup and job processing times are controllable by allocating a continuously divisible nonrenewable resource. They present polynomial time algorithms to find the job sequence, the partition of the job sequence into batches and the resource allocation, which minimize the total completion time or the total production cost (inventory plus resource costs). Yin *et al.* [19] consider the problem of single-machine batch delivery scheduling with an assignable common due date and controllable processing times. The job processing time is either a linear or a convex function of the amount of a continuously divisible and non-renewable resource allocated to the job. Finished jobs are delivered in batches and there is no capacity limit on each delivery batch. The objective is to find a job sequence, a partition of the job sequence into batches, a common due date, and resource allocation that jointly minimise a cost function based on earliness, weighted number of tardy jobs, job holding, due-date assignment, batch delivery, makespan, and resource consumption. They provide some properties of the optimal solution, and show that both the problems with the linear and convex resource consumption functions can be solved in polynomial time.

In addition, scheduling has been extensively studied in the literature often under the assumption that all the jobs are processed. However, in many cases the scheduler may need to decide whether or not to accept the jobs and to efficiently schedule the set of accepted jobs on the machines. Job rejection may be considered, *e.g.*, in the case where the machine capacity is too low, which does not allow the scheduler to process all the jobs. In such a case, the rejected jobs may be either outsourced or not get served at all, thus incurring a rejection penalty due to either the outsourcing cost or the loss in income and reputation (see [12]). The idea of scheduling with rejection is relatively new. It was first introduced by Bartal *et al.* [2], who study the problem of minimizing the sum of makespan and rejection penalties on a set of identical parallel machines, focusing on approximations. Due to the importance of scheduling with rejection strategy, some researchers investigate this kind of scheduling problems in the framework of batch scheduling under batch availability. Shabtay [11] study a scheduling problem with rejection on a single serial batching machine, where the objectives are to minimize the total completion time and the total rejection cost. They consider four different problem variations. The first is to minimize the sum of the two objectives. The second and the third are to minimize one objective, given an upper bound on the value of the other objective and the last is to find a Pareto-optimal solution for each Pareto-optimal point. They provide a polynomial time procedure to solve the first variation, and show that the three other variations are NP-hard and construct a pseudo-polynomial time algorithm for solving the NP-hard problems. Yin *et al.* [18] provide an alternative algorithm to solve the first variant and a FPTAS for the fourth variant investigated in Shabtay [11], which are more efficient than those developed by Shabtay [11] from a theoretical perspective.

It is natural and interesting to study the scheduling problem in which rejected jobs, batch scheduling, and resource allocation are considered simultaneously. To the best of our knowledge, there is no research on such a problem. The aim of this paper is twofold. One is to study this more realistic and complex scheduling model. The other is to ascertain the computational complexity status and provide solution procedures, if viable, for the problems under consideration. The rest of the paper is organized as follows: Section 2 considers the problem with fixed setup and job processing times, and provide an $O(n^6)$ dynamic programming solution algorithm for it. Section 3 studies the problem with convex resource-dependent setup and processing times, and provide an $O(n^6)$ dynamic programming algorithm to solve it. We also show that the case of both problems with common setup time can be solved in $O(n^5)$ time. Section 4 concludes the paper and suggest some future research topics.

2. Optimal solution for the case with fixed setup and processing times

This section first studies the problem with fixed setup and processing times because it seems that this problem has not been studied in the literature. Given any feasible resource allocations $\mathbf{u}$ and $\mathbf{v}$, the setup and job processing times, and the resource consumption cost are fixed. So the problem in this case reduces to finding an accepted job sequence S, a rejected job set $\overline{A}$, and a partition of the job sequence S into bathes $\mathbf{B}$ that jointly minimize

$$Z(S,\overline{A},\mathbf{B}) = \sum_{j=1}^{n_a}(\alpha D_{[j]} + \beta H_{[j]}) + \sum_{j\in\overline{A}} e_j, \tag{2.1}$$

denoted by $1|rej,\mathbf{B}|\sum_{j=1}^{n_a}(\alpha D_{[j]} + \beta H_{[j]}) + \sum_{j\in\overline{A}} e_j$.

It is easy to derive that the optimal schedule will not include idle times since removing any idle time will reduce the objective value. For a fixed job sequence $\pi = (S,\overline{A})$ with n_a accepted jobs in the order of S, assume that the job sequence $S = (J_{[1]},\ldots,J_{[n_a]})$ is partitioned into m batches $B_1 = \{J_{[1]},\ldots,J_{[l_1]}\}$, $B_2 = \{J_{[l_1+1]},\ldots,J_{[l_2]}\},\ldots,$ and $B_m = \{J_{[l_{m-1}+1]},\ldots,J_{[l_m]}\}$, where l_i denotes the number of jobs in the first i batches for $i = 0,1,\ldots,m$, with $l_0 = 0$ and $l_m = n_a$. So

$$D_{[j]} = \sum_{l=1}^{i} s_l + \sum_{l=1}^{l_i} p_{[l]} \tag{2.2}$$

and

$$H_{[j]} = D_j - C_j = D_{[j]} - \left(\sum_{l=1}^{i} s_l + \sum_{l=l_{i-1}+1}^{j} p_{[l]}\right) = \sum_{l=j+1}^{l_i} p_{[l]} \tag{2.3}$$

for $j \in \{J_{[l_{i-1}+1]},\ldots,J_{[l_i]}\}$ and $i = 1,\ldots,m$. Denote the number of jobs in B_i by $|B_i|$, *i.e.*, $|B_i| = l_i - l_{i-1}$ for $i = 1,\ldots,m$. For the fixed job sequence and partition, equation (4) can be re-formulated as

$$\begin{aligned}
Z(S,\overline{A},\mathbf{B}) &= \alpha\sum_{i=1}^{m}(l_i - l_{i-1})D_j + \beta\sum_{j=1}^{n_a} H_j + \sum_{J_j\in\overline{A}} e_j \\
&= \alpha\sum_{i=1}^{m}(l_i - l_{i-1})\left(\sum_{l=1}^{i} s_l + \sum_{l=1}^{l_i} p_{[l]}\right) + \beta\sum_{i=1}^{m}\sum_{j=l_{i-1}}^{l_i} H_j + \sum_{J_j\in\overline{A}} e_j \\
&= \alpha\sum_{i=1}^{m}(l_i - l_{i-1})\sum_{l=1}^{i}(s_l + P_l) + \beta\sum_{i=1}^{m}\sum_{j=l_{i-1}+1}^{l_i}\sum_{l=j+1}^{l_i} p_{[l]} + \sum_{J_j\in\overline{A}} e_j \\
&= \alpha\sum_{i=1}^{m}(n_a - l_{i-1})(s_i + P_i) + \beta\sum_{i=1}^{m}\sum_{j=l_{i-1}+1}^{l_i}(j - 1 - l_{i-1})p_{[j]} + \sum_{J_j\in\overline{A}} e_j,
\end{aligned} \tag{2.4}$$

where $P_i = \sum_{j=l_{i-1}+1}^{l_i} p_{[j]}$ denotes the total processing time of the jobs in the ith batch for $i = 1,\ldots,m$.

In what follows, borrowing the idea from Yin *et al.* [17,19], some structural properties of an optimal solution for the problem $1|rej,\mathbf{B}|\sum_{j=1}^{n_a}(\alpha D_{[j]} + \beta H_{[j]}) + \sum_{j\in\overline{A}} e_j$ are presented.

Lemma 2.1. *For the problem* $1|rej, \boldsymbol{B}|\sum_{j=1}^{n_a}(\alpha D_{[j]}+\beta H_{[j]})+\sum_{j\in\overline{A}} e_j$, *an optimal schedule exists in which the jobs in any batch are sequenced in nonincreasing order of* p_j, i.e., *in the longest processing time (LPT) order.*

Proof. The optimal job sequence within each batch should minimize the second term in equation (2.4), *i.e.*, the total holding cost, since it is the only term within equation (2.4) that is dependent on the internal sequence within the batch. The total holding cost of the jobs in batch i is $\beta\sum_{j=l_{i-1}+1}^{l_i}(j-1-l_{i-1})p_{[j]}$ for $i=1,\ldots,m$. The result follows from the well-known result in linear algebra about the minimization of the scalar product of two vectors (see Hardy *et al.* [5]). □

Lemma 2.2. *For the problem* $1|rej, \boldsymbol{B}|\sum_{j=1}^{n_a}(\alpha D_{[j]}+\beta H_{[j]})+\sum_{j\in\overline{A}} e_j$, *an optimal schedule exists in which the jobs in any two consecutive batches* B_l *and* B_{l+1} *satisfy* $p_i \le p_j$ *if* J_i *is in* B_l *and* J_j *is in* B_{l+1}.

Proof. Because of Lemma 2.1, the condition of Lemma 2.2 holds for an optimal schedule if and only if $p_i \le p_j$, where J_i is the first job in B_l and J_j is the last job in B_{l+1}. Now suppose that there is an optimal schedule $\pi=(S,\overline{A})$ with n_a accepted jobs, and two consecutive batches B_l and B_{l+1} in which $p_i > p_j$, where J_i is the first job in B_l and J_j is the last job in B_{l+1}. Now break down the proof into two cases.

Case 1: $|B_l| > |B_{l+1}|-1$. For this case, consider a new schedule $\pi'=(S',\overline{A})$ constructed by swapping jobs J_i and J_j. By equation (2.4), the difference between the objective values of schedules π' and π is

$$\begin{aligned}\Delta &= -\alpha(n_a-l_{l-1})(p_i-p_j)+\alpha(n_a-l_l)(p_i-p_j)\\ &\quad +\beta(|B_{l+1}|-1)(p_i-p_j)\\ &= \alpha(-l_l+l_{l-1})(p_i-p_j)+\beta(|B_{l+1}|-1)(p_i-p_j)\\ &= (\beta(|B_{l+1}|-1)-\alpha|B_l|)(p_i-p_j),\end{aligned}$$

which is negative because $\alpha\ge\beta$, $|B_l|>|B_{l+1}|-1$, and $p_i>p_j$.

Case 2: $|B_l| \le |B_{l+1}|-1$. For this case, consider a new schedule $\pi'=(S',\overline{A})$ constructed by placing job J_j in the last position of B_l. By equation (2.4), the difference between the objective values of schedules π' and π is

$$\begin{aligned}\Delta &= \alpha(n_a-l_{l-1})(s_l+P_l+p_j)+\alpha(n_a-l_l-1)(s_{l+1}+P_{l+1}-p_j)\\ &\quad +\beta|B_l|p_j-\beta(|B_{l+1}|-1)p_j-\alpha(n_a-l_{l-1})(s_l+P_l)\\ &\quad -\alpha(n_a-l_l)(s_{l+1}+P_{l+1})\\ &= \alpha(n_a-l_{l-1})p_j-\alpha P_{l+1}(S)-\alpha(n_a-l_l-1)p_j\\ &\quad +\beta|B_l|p_j-\beta(|B_{l+1}|-1)p_j-\alpha s_{l+1}\\ &\le \alpha n_a p_j-\alpha l_{l-1}p_j-\alpha|B_{l+1}|p_j-\alpha(n_a-l_k-1)p_j\\ &\quad +\beta|B_k|p_j-\beta(|B_{k+1}|-1)p_j-\alpha s_{l+1}\\ &= (\alpha+\beta)((|B_l|+1)-|B_{l+1}|)p_j-\alpha s_{l+1},\end{aligned}$$

which is negative because $|B_l|\le|B_{l+1}|-1$.

Thus, in both cases, schedule π' is at least as good as π. Repeating this modifying argument for all the accepted jobs not sequenced in the specified order yields the result. □

The following result presents the properties of the batch sizes.

Lemma 2.3. *For the problem* $1|rej, \boldsymbol{B}|\sum_{j=1}^{n_a}(\alpha D_{[j]}+\beta H_{[j]})+\sum_{j\in\overline{A}} e_j$, *an optimal schedule exists in which* $|B_l|\ge|B_{l+1}|$ *for any two consecutive batches* B_l *and* B_{l+1}.

Proof. Given an optimal schedule $\pi = (S, \overline{A})$ with two consecutive batches B_l and B_{l+1} in S such that $|B_l| < |B_{l+1}|$. By the proof of Case 2 in Lemma 2.2, moving one job from B_{l+1} into B_l does not increase the objective value. Repeating this modifying argument until $|B_l| \geq |B_{l+1}|$ yields the result. □

Lemma 2.4. *For a given n_a value, an optimal schedule for the problem $1|rej, \boldsymbol{B}| \sum_{j=1}^{n_a} (\alpha D_{[j]} + \beta H_{[j]}) + \sum_{j \in \overline{A}} e_j$ exists in which*

$$|B_l| \leq \left\lfloor \sqrt{\frac{4\alpha(n_a - 1)s_{\max}^{n_a}}{(\alpha + \beta)p_{\min}} + 1} \right\rfloor$$

for any batch B_l, where $p_{\min} = \min\{p_1, \ldots, p_n\}$, $s_{\max}^{n_a} = \max\{s_1, \ldots, s_{n_a}\}$, and $\lfloor x \rfloor$ denotes the largest integer less than or equal to x.

Proof. Suppose that there exists an optimal schedule $\pi = (S, \overline{A})$ with n_a accepted jobs and a batch B_l for which $|B_l| > \left\lfloor \sqrt{\frac{4\alpha(n_a-1)s_{\max}^{n_a}}{(\alpha+\beta)p_{\min}} + 1} \right\rfloor$. Clearly, $|B_l| \geq 2$. Let $q = \frac{|B_l|}{2}$ if $|B_l|$ is even, and $q = \frac{|B_l|-1}{2}$ otherwise. Now consider a new schedule $\pi' = (S', \overline{A})$ constructed by splitting batch B_l into two batches $\{J_{[l_{l-1}+q+2]}, J_{[l_{l-1}+q+3]}, \ldots, J_{[l_l]}\}$ and $\{J_{[l_{l-1}+1]}, J_{[l_{l-1}+2]}, \ldots, J_{[l_{l-1}+q+1]}\}$. By equation (2.4), the difference between the objective values of schedules π' and π is

$$\begin{aligned}
\Delta &= \alpha(n_a - l_{l-1}) \left(s_l + \sum_{j=l_{l-1}+q+1}^{l_l} p_{[j]} \right) \\
&\quad + \alpha(n_a - (l_l - q)) \left(s_{l+1} + \sum_{j=l_{l-1}+1}^{l_{l-1}+q} p_{[j]} \right) \\
&\quad - \alpha(n_a - l_{l-1}) \left(s_l + \sum_{j=l_{l-1}+1}^{l_l} p_{[j]} \right) - \beta q \sum_{j=l_{l-1}+v+1}^{l_l} p_{[j]} \\
&\quad + \alpha \sum_{j=l+1}^{m} (n_a - l_{j-1})(s_{j+1} - s_j) \\
&= -\alpha(|B_l| - q) \sum_{j=l_{l-1}+1}^{l_{l-1}+q} p_{[j]} - \beta q \sum_{j=l_{l-1}+q+1}^{l_l} p_{[j]} \\
&\quad + \alpha(n_a - (l_l - q))s_{l+1} + \alpha \sum_{j=l+1}^{m} (l_j - l_{j-1})s_{j+1} \\
&\quad - \alpha(n_a - l_l)s_{l+1} \\
&= -\alpha(|B_l| - q) \sum_{j=l_{l-1}+1}^{l_{l-1}+q} p_{[j]} - \beta q \sum_{j=l_{l-1}+q+1}^{l_l} p_{[j]} \\
&\quad + \alpha q s_{l+1} + \alpha \sum_{j=l+1}^{m} |B_j| s_{j+1} \\
&\leq -(\alpha + \beta)(|B_l| - q)q p_{\min} + \alpha(q + |B_{l+1}| + \ldots + |B_m|)s_{\max}^{n_a} \\
&\leq -(\alpha + \beta)(|B_l| - q)q p_{\min} + \alpha(n_a - 1)s_{\max}^{n_a}.
\end{aligned}$$

Now, if $|B_l|$ is even, then $(|B_l|-l)l = \frac{|B_l|^2}{4}$; otherwise, $(|B_l|-q)q = \frac{|B_l|^2-1}{4}$. Since $|B_l| > \left\lfloor \sqrt{\frac{4\alpha(n_a-1)s_{\max}^{n_a}}{(\alpha+\beta)p_{\min}} + 1} \right\rfloor$, one have $(\alpha+\beta)(|B_l|-q)qp_{\min} \geq \frac{(\alpha+\beta)(|B_l|^2-1)p_{\min}}{4} > \alpha(n_a-1)s_{\max}^{n_a}$, implying that $\Delta < 0$. Therefore, π' is a better schedule than π, as required. □

In what follows, number the jobs in the shortest processing time (SPT) order so that $p_1 \leq p_2 \leq \ldots \leq p_n$. This numbering can be done in $O(n \log n)$ time. Next, a recursion relation will be provided that can be exploited to design a polynomial-time dynamic programming algorithm for the problem $1|rej, \mathbf{B}| \sum_{j=1}^{n_a}(\alpha D_{[j]} + \beta H_{[j]}) + \sum_{j \in \overline{A}} e_j$.

Let us begin with introducing some notation that will be used later:

$(i,j,x,y,r)_{n_a}$: a state representing the situation where the jobs $\{J_1,\ldots,J_i\}$ have been scheduled, provided that in the final optimal schedule for the whole job set N, there are exactly n_a accepted jobs, and that in the current partial schedule for jobs $\{J_1,\ldots,J_i\}$, there are j ($j \leq \min\{i, n_a\}$) accepted jobs, and there are y jobs in the the current last batch, which will be the rth batch ($r = j = 0$ or $1 \leq r \leq j$) and contains exactly x jobs in the final optimal schedule for the whole job set N.

$F_{n_a}(i,j,x,y,r)$: the optimal solution value of any schedule in state $(i,j,x,y,r)_{n_a}$.

$S_{n_a}(i,j,x,y,r)$: any schedule in state $(i,j,x,y,r)_{n_a}$ with the solution value $F_{n_a}(i,j,x,y,r)$.

$\phi_{n_a}(i,j,x,y,r)$: a variable, *i.e.*, $\phi_{n_a}(i,j,x,y,r)$ is equal to the position number of job J_i in the final optimal schedule for the whole job set N if job J_i is an accepted job, otherwise $\phi_{n_a}(i,j,x,y,r) = \emptyset$, provided that the jobs $\{J_1,\ldots,J_i\}$ are scheduled corresponding to $F_{n_a}(i,j,x,y,r)$. This function is used to keep track of the positions of the accepted jobs in the optimal schedule.

$\psi_{n_a}(i,j,x,y,r)$: a binary variable, *i.e.*, $\psi_{n_a}(i,j,x,y,r) = 1$ if job J_i is the last job in the current last batch corresponding to $F_{n_a}(i,j,x,y,r)$, and $\psi_{n_a}(i,j,x,y,r) = 0$ otherwise. This function is used to indicate where batches are formed in the optimal schedule.

By definition, set $F_{n_a}(i,j,x,y,r) = +\infty$ if no such schedule exists. Then the schedule $S_{n_a}(i,j,x,y,r)$ must have been constructed by taking one of the following three decisions in the previous state.

(1) Job J_i is a rejected job. In this case, $S_{n_a}(i,j,x,y,r)$ must have been obtained from schedule $S_{n_a}(i-1,j,x,y,r)$ and $F_{n_a}(i,j,x,y,r) = F_{n_a}(i-1,j,x,y,r) + e_i$.

(2) Job J_i is an accepted job and assigned to the last batch currently containing at least $y \geq 2$ jobs. In this case, $S_{n_a}(i,j,x,y,r)$ must have been obtained from schedule $S_{n_a}(i-1,j-1,x,y-1,r)$; job J_i is currently the first job in the current last batch, but it will be the $(x-y+1)$th job in this batch in the final optimal schedule for N; and there are $j-y$ jobs processed before the current last batch in the final optimal schedule for N, implying that $\phi_{n_a}(i,j,x,y,r) = j+x-2y+1$. Hence the contribution made by job J_i to the objective function of the scheduling problem is $(\alpha(n_a-i+j+y)+\beta(x-y))p_i$, so $F_{n_a}(i,j,x,y,r) = F_{n_a}(i-1,j-1,x,y-1,r) + (\alpha(n_a-j+y)+\beta(x-y))p_i$.

(3) Job J_i is assigned to the last batch currently containing only one job, *i.e.*, J_i is currently the sole job in the current last batch. In this case, $S_{n_a}(i,j,x,1,r)$ must have been obtained from schedule $S_{n_a}(i-1,j-1,z,z,r-1)$ with the minimum solution value $F_{n_a}(i-1,j-1,z,z,r-1)$ for $z \geq x$ by Lemma 2.3; job J_i will be the xth job in this batch in the final optimal schedule for N; and there are $j-1$ jobs processed before the current last batch in the final optimal schedule for N, so $\phi_{n_a}(i,j,x,1,r) = j+x-1$. Hence, the contribution of job J_i to the objective function of the scheduling problem is $(\alpha(n_a-i+j+1)+\beta(x-1))p_i$, so $F_{n_a}(i,j,x,y,r) = \min_{z \geq x}\{F_{n_a}(i-1,j-1,z,z,r-1)\} + \alpha(n_a-j+1)(s_r+p_i) + \beta(x-1)p_i$.

Based on the above analysis, an algorithm for the problem is given as follows:

Algorithm 1.

Denote $\overline{B}_{n_a} = \left\lfloor \sqrt{\frac{4\alpha(n_a-1)s_{\max}^{n_a}}{(\alpha+\beta)p_{\min}} + 1} \right\rfloor$ for $n_a = 1,\ldots,n$. Re-number the jobs in the SPT order such that $p_1 \leq p_2 \leq \ldots \leq p_n$.

(1) Initial conditions

(a) If (n_a, i, j, x, y, r) does not satisfy $0 \le n_a \le n$, $1 \le i \le n$, $0 \le j \le \min\{i, n_a\}$, $r = j = 0$ or $1 \le r \le j$, $0 \le x \le \min\{n_a, \overline{B}_{n_a}\}$, and $0 \le y \le \min\{j, x\}$, then

$$F_{n_a}(i, j, x, y, r) = +\infty, \phi_{n_a}(i, j, x, y, r) = \emptyset, \psi_{n_a}(i, j, x, y, r) = \emptyset.$$

(b) If $x = r = 0$ and $n_a < n$, then

$$F_{n_a}(1, 0, 0, 0, 0) = e_1,$$
$$\phi_{n_a}(1, 0, 0, 0, 0) = \phi,\ \psi_{n_a}(1, 0, 0, 0, 0) = 0.$$

(c) If $1 \le x \le \min\{n_a, \overline{B}_{n_a}\}$ and $n_a \ge 1$, then

$$F_{n_a}(1, 1, x, 1, 1) = \alpha n_a(s_1 + p_1) + \beta(x - 1)p_1,$$
$$\phi_{n_a}(1, 1, x, 1, 1) = x,\ \psi_{n_a}(1, 1, x, 1, 1) = 1.$$

(2) Recursive relations

For $0 \le n_a \le n$, $2 \le i \le n$, $0 \le j \le \min\{i, n_a\}$, $r = j = 0$ or $1 \le r \le j$, and $0 \le x \le \min\{n_a, \overline{B}_{n_a}\}$:

(a) For $2 \le y \le \min\{j, x\}$:

$$F_{n_a}(i, j, x, y, r) = \min \begin{cases} F_{n_a}(i-1, j, x, y, r) + e_i,\ (\equiv X) \\ F_{n_a}(i-1, j-1, x, y-1, r) \\ \quad +(\alpha(n_a - j + y) + \beta(x - y))p_i,\ (\equiv Y) \end{cases}$$
$$\phi_{n_a}(i, j, x, y, r) = \begin{cases} \phi, & \text{if } F_{n_a}(i, j, x, y, r) = X, \\ j + x - 2y + 1, & \text{if } F_{n_a}(i, j, x, y, r) = Y, \end{cases}$$
$$\psi_{n_a}(i, j, x, y, r) = 0.$$

(b) For $y = 1$:

$$F_{n_a}(i, j, x, 1, r) = \min \begin{cases} F_{n_a}(i-1, j, x, y, r) + e_i,\ (\equiv U) \\ \min_{z \ge x}\{F_{n_a}(i-1, j-1, z, z, r-1)\} \\ \quad +\alpha(n_a - j + 1)(s_r + p_i) + \beta(x - 1)p_i,\ (\equiv V) \end{cases}$$
$$\phi_{n_a}(i, j, x, 1, r) = \begin{cases} \phi, & \text{if } F_{n_a}(i, j, x, 1, r) = U, \\ j + x - 1, & \text{if } F_{n_a}(i, j, x, 1, r) = V, \end{cases}$$
$$\psi_{n_a}(i, j, x, 1, r) = \begin{cases} 0, & \text{if } F_{n_a}(i, j, x, 1, r) = U, \\ 1, & \text{if } F_{n_a}(i, j, x, 1, r) = V. \end{cases}$$

(3) The optimal solution value is $F^* = \min\{F_{n_a}(n, n_a, x, x, r) | 0 \le n_a \le n, 0 \le x \le \min\{n_a, \overline{B}_{n_a}\}, r = n_a = 0 \text{ or } 1 \le r \le n_a\}$.

Let u^*, x^*, and r^* be the corresponding values. The optimal sequence can be obtained by recursively searching the position functions $\phi_{n_a^*}(i, j, x, y, r)$ beginning with $\phi_{n_a^*}(n, n_a^*, x^*, x^*, r^*)$. The optimal batch delivery date of each job can be determined from the batch indicator functions $\psi_{n_a^*}(i, j, x, y, r)$.

Theorem 2.5. *The problem* $1|rej, \boldsymbol{B}| \sum_{j=1}^{n_a}(\alpha D_{[j]} + \beta H_{[j]}) + \sum_{j \in \overline{A}} \mathrm{e}_j$ *can be solved in* $O(n^6)$ *time by Algorithm* 1.

TABLE 1. The job processing times and rejection costs.

j	1	2	3	4
p_j	3	4	7	9
e_j	17	15	20	25

Proof. Optimality is guaranteed by Lemmas 2.1–2.4 and the principles underlying dynamic programming. Now, consider the computational complexity of the dynamic programming solution algorithm. There are at most n^6 states when $y \geq 2$ and at most n^5 states when $y = 1$. Computing each $F_{n_a}(i, j, x, y, r)$ requires constant time when $y \geq 2$, while computing each $F_{n_a}(i, j, x, 1, r)$ requires $O(n)$ time when $y = 1$, so computing all of them requires $O(n^6)$ time. Computing F^* requires linear time. The overall computational complexity of the algorithm is therefore $O(n^6)$. □

Note that in the case of a constant setup time (*i.e.*, $s_i = s$, $i = 1, \ldots, m$), which implies that the setup is batch-independent, one can drop the decision variable r in Algorithm 1 and obtain the following result.

Theorem 2.6. *The problem* $1|rej, \boldsymbol{B}| \sum_{j=1}^{n_a} (\alpha D_{[j]} + \beta H_{[j]}) + \sum_{j \in \overline{A}} \mathrm{e}_j$ *can be solved in* $O(n^5)$ *time in the case with a common setup time.*

The following example is presented to illustrate Algorithm 1 for the problem $1|rej, \mathbf{B}| \sum_{j=1}^{n_a} (\alpha D_{[j]} + \beta H_{[j]}) + \sum_{j \in \overline{A}} \mathrm{e}_j$.

Example 2.7. Let $\alpha = 1$ and $\beta = 2$. Consider a problem containing $n = 4$ jobs with $s = 3$ and the other corresponding parameters shown in Table 1. Now we can solve the problem $1|rej, \mathbf{B}| \sum_{j=1}^{n_a} (\alpha D_{[j]} + \beta H_{[j]}) + \sum_{j \in \overline{A}} \mathrm{e}_j$.

In the sequel, we apply Algorithm 1 to solve this numerical example.

Step 1. The jobs are already indexed in the SPT order.

Step 2. Calculate $\overline{B_1} = 1$, $\overline{B_2} = \left\lfloor \sqrt{\frac{4\times 1}{3} + 1} \right\rfloor = 1$, $\overline{B_2} = \left\lfloor \sqrt{\frac{4\times 2}{3} + 1} \right\rfloor = 1$, and $\overline{B_4} = \left\lfloor \sqrt{\frac{4\times 3}{3} + 1} \right\rfloor = 2$. Set $F_{n_a}(1, 0, 0, 0) = e_1 = 17$ for $n_a = 0, 1, \ldots, 3$, and $F_{n_a}(1, 1, 1, 1) = \alpha n_a(s+p_1)+\beta(1-1)p_1 = 6n_a$ for $n_a = 1, \ldots, 3$ and $F_4(1, 1, x, 1) = \alpha n_a(s + p_1) + \beta(x - 1)p_1 = 24 + 6(x - 1)$ for $x = 1, 2$.

Step 3.

For $n_a = 0$, we have

$F_0(1, 0, 0, 0) = e_1 = 17$,
$F_0(2, 0, 0, 0) = 17 + e_2 = 32$,
$F_0(3, 0, 0, 0) = 32 + e_3 = 52$,
$F_0(4, 0, 0, 0) = 52 + e_4 = 77$.

For $n_a = 1$, we have

For $j = 2$, one have
$F_1(2, 0, 0, 0) = 17 + e_2 = 32$,
$F_1(2, 1, 1, 1) = F_1(1, 0, 0, 0) + \alpha n_a(s + p_2) + \beta(1 - 1)p_2 = 17 + 7 = 24$,
$F_1(3, 0, 0, 0) = 32 + e_3 = 52$,
$F_1(3, 1, 1, 1) = \min\{F_1(2, 0, 0, 0) + \alpha n_a(s + p_3), F_1(2, 1, 1, 1) + e_3\} = \min\{42, 44\} = 42$,
$F_1(4, 1, 1, 1) = \min\{F_1(3, 0, 0, 0) + \alpha n_a(s + p_4), F_1(3, 1, 1, 1) + e_4\} = \min\{64, 67\} = 64$.

For $n_a = 2$, we have

For $j = 2$, one have
$F_2(2, 0, 0, 0) = 17 + e_2 = 32$,

$F_2(2,1,1,1) = F_2(1,0,0,0) + \alpha(n_a - 1 + 1)(s + p_2) = 17 + 14 = 31,$

$F_2(2,2,1,1) = F_2(1,1,1,1) + \alpha(n_a - 2 + 1)(s + p_2) = 6 \times 2 + 7 = 19,$

$F_2(3,0,0,0) = 32 + e_3 = 52,$

$F_2(3,1,1,1) = \min\{F_2(2,0,0,0) + \alpha(n_a - 1 + 1)(s + p_3), F_2(2,1,1,1) + e_3\} = \min\{52, 51\} = 51,$

$F_2(3,2,1,1) = \min\{F_2(2,1,1,1) + \alpha(n_a - 2 + 1)(s + p_3), F_2(2,2,1,1) + e_3\} = \min\{41, 39\} = 39,$

$F_2(4,2,1,1) = \min\{F_2(3,1,1,1) + \alpha(n_a - 2 + 1)(s + p_4), F_2(3,2,1,1) + e_4\} = \min\{63, 64\} = 63.$

For $n_a = 3$, we have

For $j = 2$, one have

$F_3(2,0,0,0) = 17 + e_2 = 32,$

$F_3(2,1,1,1) = F_3(1,0,0,0) + \alpha(n_a - 1 + 1)(s + p_2) = 17 + 14 = 39,$

$F_3(2,2,1,1) = F_3(1,1,1,1) + \alpha(n_a - 2 + 1)(s + p_2) = 6 \times 3 + 14 = 32,$

$F_3(3,1,1,1) = \min\{F_3(2,0,0,0) + \alpha(n_a - 1 + 1)(s + p_3), F_3(2,1,1,1) + e_3\} = \min\{62, 59\} = 59,$

$F_3(3,2,1,1) = \min\{F_3(2,1,1,1) + \alpha(n_a - 2 + 1)(s + p_3), F_3(2,2,1,1) + e_3\} = \min\{49, 52\} = 49,$

$F_3(3,3,1,1) = F_3(2,2,1,1) + \alpha(n_a - 3 + 1)(s + p_3) = 42,$

$F_3(4,3,1,1) = \min\{F_3(3,2,1,1) + \alpha(n_a - 3 + 1)(s + p_4), F_3(3,3,1,1) + e_4\} = \min\{61, 67\} = 61.$

For $n_a = 4$, we have

For $j = 2$, one have

$F_4(2,2,1,1) = F_4(1,1,1,1) + \alpha(n_a - 2 + 1)(s + p_2) = 24 + 21 = 45,$

$F_4(2,2,2,2) = F_4(1,1,2,1) + \alpha(n_a - 2 + 2)p_2 + \beta(2 - 2)p_2 = 30 + 16 = 46,$

$F_4(3,3,1,1) = \min\{F_3(2,2,1,1) + \alpha(n_a - 3 + 1)(s + p_3), F_3(2,2,2,2) + \alpha(n_a - 3 + 1)(s + p_3)\} = \min\{55, 56\} = 55,$

$F_4(3,3,2,1) = F_4(2,2,2,2) + \alpha(n_a - 3 + 1)(s + p_3) + \beta(2 - 1)p_3 = 46 + 20 + 14 = 80,$

$F_4(4,4,1,1) = F_4(3,3,1,1) + \alpha(n_a - 4 + 1)(s + p_4) = 55 + 12 = 67,$

$F_4(4,4,2,2) = F_4(3,3,2,1) + \alpha(n_a - 4 + 2)p_4 + \beta(2 - 2)p_4 = 80 + 18 = 98.$

Step 4. The optimal solution value is $F^* = 61$ which corresponds to the state $S_3(4,3,1,1)$. By backtracking, the set of accepted jobs is $A = \{J_2, J_3, J_4\}$ and each job is processed in a single batch such that there are three batches: $B_1 = \{J_2\}$, $B_2 = \{J_3\}$ and $B_3 = \{J_4\}$.

3. Optimal solution for the case with convex resource consumption functions

This section provides an algorithm to solve the problem $1|covx, rej, \mathbf{B}|\sum_{j=1}^{n_a}(\alpha D_{[j]} + \beta H_{[j]} + \delta_{[j]} v_{[j]}) + \sum_{i=1}^{m} \gamma_i u_i + \sum_{j \in \overline{A}} e_j$ with the resource consumption functions given in equations (1.1) and (1.2).

Consider again an arbitrary fixed job sequence $\pi = (S, \overline{A})$ with n_a accepted jobs in the order of S. Assume that the job sequence $S = (J_{[1]}, \ldots, J_{[n_a]})$ is partitioned into m batches $B_1 = \{J_{[1]}, \ldots, J_{[l_1]}\}$, $B_2 = \{J_{[l_1+1]}, \ldots, J_{[l_2]}\}, \ldots,$ and $B_m = \{J_{[l_{m-1}+1]}, \ldots, J_{[l_m]}\}$, where l_i denotes the number of jobs in the first i batches for $i = 0, 1, \ldots, m$, with $l_0 = 0$ and $l_m = n_a$. By substituting the convex resource functions in equations (1.1) and (1.2) for each u_i and v_j into the objective function in equation (1.3), one obtain from the deduction

of equation (2.4) that

$$\begin{aligned} Z(S,\overline{A},\mathbf{B}) &= \alpha\sum_{i=1}^{m}(n_a-l_{i-1})\left(\left(\frac{\omega_i}{u_i}\right)^k+P_i\right)\\ &\quad+\beta\sum_{i=1}^{m}\sum_{j=l_{i-1}+1}^{l_i}(j-1-l_{i-1})\left(\frac{w_{[j]}}{v_{[j]}}\right)^k\\ &\quad+\sum_{i=1}^{m}\gamma_i\left(\frac{\omega_i}{u_i}\right)^k+\sum_{j=1}^{n_a}\delta_{[j]}\left(\frac{w_{[j]}}{v_{[j]}}\right)^k+\sum_{J_j\in\overline{A}}e_j\\ &= \alpha\sum_{i=1}^{m}(n_a-l_{i-1})\left(\frac{\omega_i}{u_i}\right)^k+\sum_{i=1}^{m}\sum_{j=l_{i-1}+1}^{l_i}(\alpha(n_a-l_{i-1})\\ &\quad+\beta(j-1-l_{i-1}))\left(\frac{w_{[j]}}{v_{[j]}}\right)^k+\sum_{i=1}^{m}\gamma_i\left(\frac{\omega_i}{u_i}\right)^k\\ &\quad+\sum_{j=1}^{n_a}\delta_{[j]}\left(\frac{w_{[j]}}{v_{[j]}}\right)^k+\sum_{J_j\in\overline{A}}e_j, \end{aligned} \tag{3.1}$$

where $P_i=\sum_{j=l_{i-1}+1}^{l_i}\left(\frac{w_{[j]}}{v_{[j]}}\right)^k$ denotes the total processing time of the jobs in the ith batch for $i=1,\ldots,m$.

The following lemma determines the optimal resource allocation, denoted by $(\mathbf{u}^*,\mathbf{v}^*)$, as a function of the job sequence π, the number of accepted jobs n_a, and the partition of the accepted job sequence into batches $\mathbf{B}$.

Lemma 3.1. *For the problem* $1|covx,rej,\boldsymbol{B}|\sum_{j=1}^{n_a}(\alpha D_{[j]}+\beta H_{[j]}+\delta_{[j]}v_{[j]})+\sum_{i=1}^{m}\gamma_i u_i+\sum_{j\in\overline{A}}e_j$, *the optimal resource allocation* $(\boldsymbol{u}^*,\boldsymbol{v}^*)$ *as a function of the job sequence* π, *the number of accepted jobs* n_a, *and the partition of the accepted job sequence into batches* $\boldsymbol{B}$ *is*:

$$u_i^*=\left(\frac{k\alpha(n_a-l_{i-1})}{\gamma_i}\right)^{\frac{1}{k+1}}\times\omega_i^{\frac{k}{k+1}} \tag{3.2}$$

for $i=1,\ldots,m$, *and*

$$v_{[j]}^*=\left(\frac{k(\alpha(n_a-l_{i-1})+\beta(j-1-l_{i-1}))}{\delta_{[j]}}\right)^{\frac{1}{k+1}}\times w_{[j]}^{\frac{k}{k+1}} \tag{3.3}$$

for $i=1,\ldots,m;j=l_{i-1}+1,\ldots,l_i$.

Proof. Differentiating each of the terms of the objective in equation (3.1) with respect to u_i and $v_{[j]}$ for $i=1,2,\ldots,m$ and $j=1,2,\ldots,n$, respectively, equating it to zero, and solving it for u_i and $v_{[j]}$, one obtain equations (3.2) and (3.3). Since the objective is a convex function with respect to u_i and $v_{[j]}$ for $i=1,2,\ldots,m$ and $j=1,2,\ldots,n$, respectively, equations (3.2) and (3.3) provide the necessary and sufficient conditions for optimality. □

Substituting equations (3.2) and (3.3) into equation (3.1), a new expression is obtained for the cost function under an optimal resource allocation as a function of the job sequence π, the number of accepted jobs n_a, and the partition of the accepted job sequence into batches $\mathbf{B}$ as follows:

$$Z(\pi,\mathbf{B},(\mathbf{u}^*,\mathbf{v}^*),n_a)=\overline{k}\sum_{i=1}^{m}\theta_i\sigma_i+\overline{k}\sum_{j=1}^{n_a}\vartheta_{[j]}\eta_j+\sum_{J_j\in\overline{A}}e_j, \tag{3.4}$$

where

$$\overline{k} = k^{\frac{-k}{k+1}} + k^{\frac{1}{k+1}}, \tag{3.5}$$

$$\theta_i = (\omega_i \gamma_i)^{\frac{k}{k+1}}, \ i = 1, \ldots, m, \tag{3.6}$$

$$\vartheta_{[j]} = (w_{[j]} \delta_{[j]})^{\frac{k}{k+1}}, \ j = 1, \ldots, n_a, \tag{3.7}$$

$$\sigma_i = (\alpha(n_a - l_{i-1}))^{\frac{1}{k+1}}, \ i = 1, \ldots, m, \tag{3.8}$$

and

$$\eta_j = (\alpha(n_a - l_{i-1}) + \beta(j - 1 - l_{i-1})))^{\frac{1}{k+1}}, \qquad i = 1, \ldots, m; j = l_{i-1} + 1, \ldots, l_i. \tag{3.9}$$

Lemma 3.2. *For the problem* $1|covx, rej, \boldsymbol{B}| \sum_{j=1}^{n_a} (\alpha D_{[j]} + \beta H_{[j]} + \delta_{[j]} v_{[j]}) + \sum_{i=1}^{m} \gamma_i u_i + \sum_{j \in \overline{A}} e_j$, *an optimal schedule exists in which the jobs in any batch are sequenced in nonincreasing order of* ϑ_j.

Proof. The proof is similar to that of Lemma 2.1. □

Lemma 3.3. *Let* $v^*_{[j]}$ *and* $\vartheta_{[j]}$ *be as given in equations* (3.3) *and* (3.7). *Then* $\vartheta_{[l]} \leq \vartheta_{[j]}$ *implies* $\left(\frac{w_{[l]}}{v^*_{[l]}}\right)^k \leq \left(\frac{w_{[j]}}{v^*_{[j]}}\right)^k$ *for any two jobs* $J_{[l]}$ *and* $J_{[j]}$ *in batch* i $(i = 1, 2, \ldots, m)$ *such that* $l_{i-1} + 1 \leq j < l \leq l_i$.

Proof. By equation (3.3) and $\vartheta_{[l]} \leq \vartheta_{[j]}$, one have

$$\begin{aligned}
\left(\frac{w_{[l]}}{v^*_{[l]}}\right)^k &= \frac{\vartheta_{[l]}}{(k(\alpha(n_a - l_{i-1}) + \beta(l - 1 - l_{i-1})))^{\frac{k}{k+1}}} \\
&\leq \frac{\vartheta_{[j]}}{(k(\alpha(n_a - l_{i-1}) + \beta(j - 1 - l_{i-1})))^{\frac{k}{k+1}}} \\
&= \left(\frac{w_{[j]}}{v^*_{[j]}}\right)^k.
\end{aligned}$$

The result follows. □

The following two results present properties regarding the sizes of the batches.

Proposition 3.4. *For the problem* $1|covx, rej, \boldsymbol{B}| \sum_{j=1}^{n_a} (\alpha D_{[j]} + \beta H_{[j]} + \delta_{[j]} v_{[j]}) + \sum_{i=1}^{m} \gamma_i u_i + \sum_{j \in \overline{A}} e_j$, *an optimal schedule exists in which* $|B_s| \geq |B_{s+1}|$ *for any two consecutive batches* B_s *and* B_{s+1}.

Proof. Consider an optimal schedule $\pi = (S, \overline{A})$ with n_a accepted jobs and two consecutive batches B_s and B_{s+1} such that $|B_s| < |B_{s+1}|$. Let J_j be the last job in B_{s+1}. Then, by Lemmas 3.2 and 3.3, $\left(\frac{w_{[l]}}{v^*_{[l]}}\right)^k \geq \left(\frac{w_j}{v^*_j}\right)^k$ for any $l = l_s + 1, l_s + 2 \ldots, l_{s+1}$. Consider a new schedule $\pi' = (S', \overline{A})$ constructed by placing job J_j in the last position of B_s without changing the setup and job processing times. By equation (3.1), the difference between

the objective values of schedules π' and π is

$$\begin{aligned}
\Delta &= \alpha(n_a - l_{s-1})\left(\left(\frac{\omega_s}{u_s^*}\right)^k + P_{[s]} + \left(\frac{w_j}{v_j^*}\right)^k\right) \\
&\quad + \alpha(n_a - l_s - 1)\left(\left(\frac{\omega_{s+1}}{u_{s+1}^*}\right)^k + P_{[s+1]} - \left(\frac{w_j}{v_j^*}\right)^k\right) \\
&\quad - \alpha(n_a - l_{s-1})\left(\left(\frac{\omega_s}{u_s^*}\right)^k + P_{[s]}\right) \\
&\quad - \alpha(n_a - l_s)\left(\left(\frac{\omega_{s+1}}{u_{s+1}^*}\right)^k + P_{[s+1]}\right) \\
&\quad + \beta|B_s|\left(\frac{w_j}{v_j^*}\right)^k - \theta(|B_{s+1}| - 1)\left(\frac{w_j}{v_j^*}\right)^k \\
&= \alpha(n_a - l_{s-1})\left(\frac{w_j}{v_j^*}\right)^k - \alpha\left(\left(\frac{\omega_{s+1}}{u_{s+1}^*}\right)^k + P_{[s+1]}\right) \\
&\quad - \alpha(n_a - l_s - 1)\left(\frac{w_j}{v_j^*}\right)^k + \beta(|B_s| - (|B_{s+1}| - 1))\left(\frac{w_j}{v_j^*}\right)^k \\
&= \alpha(|B_s| + 1)\left(\frac{w_j}{v_j^*}\right)^k - \alpha P_{[s+1]} + \theta(|B_s| - (|B_{s+1}| - 1))\left(\frac{w_j}{v_j^*}\right)^k - \alpha\left(\frac{\omega_{s+1}}{u_{s+1}^*}\right)^k \\
&\le \alpha(|B_s| + 1)\left(\frac{w_j}{v_j^*}\right)^k - \alpha|B_{s+1}|\left(\frac{w_j}{v_j^*}\right)^k \\
&\quad + \theta(|B_s| - (|B_{s+1}| - 1))\left(\frac{w_j}{v_j^*}\right)^k - \alpha\left(\frac{\omega_{s+1}}{u_{s+1}^*}\right)^k \\
&= (\alpha + \beta)((|B_s| + 1) - |B_{s+1}|)\left(\frac{w_j}{v_j^*}\right)^k - \alpha\left(\frac{\omega_{s+1}}{u_{s+1}^*}\right)^k,
\end{aligned}$$

where $P_{[s]} = \sum_{l=l_{s-1}+1}^{l_s}\left(\frac{w_{[l]}}{v_{[l]}^*}\right)^k$ and $P_{[s+1]} = \sum_{l=l_s+1}^{l_{s+1}}\left(\frac{w_{[l]}}{v_{[l]}^*}\right)^k \ge |B_{s+1}|\left(\frac{w_j}{v_j^*}\right)^k$. It follows from $|B_s| \le |B_{s+1}| - 1$ that $\Delta < 0$. Therefore, schedule π' is at least as good as π. Repeating this modifying argument until $|B_s| \ge |B_{s+1}|$ yields the result. □

Proposition 3.5. *For a given n_a value, an optimal schedule exists for the problem $1|covx, rej, \boldsymbol{B}|\sum_{j=1}^{n_a}(\alpha D_{[j]} + \beta H_{[j]} + \delta_{[j]}v_{[j]}) + \sum_{i=1}^{m}\gamma_i u_i + \sum_{j\in\overline{A}} e_j$ in which*

$$|B_s| \le \left\lfloor \sqrt{\frac{4(1+k)(\alpha n_a + \beta(n_a - 1))^{\frac{k}{k+1}}(\alpha n_a)^{\frac{1}{k+1}}\theta_{\max}^{n_a}}{(\alpha+\beta)\vartheta_{\min}}} + 1 \right\rfloor$$

for any batch B_s, where $\vartheta_{\min} = \min\{\vartheta_1, \vartheta_2, \ldots, \vartheta_n\}$ and $\theta_{\max}^{n_a} = \max\{\vartheta_1, \vartheta_2, \ldots, \vartheta_{n_a}\}$.

Proof. Suppose that there is an optimal schedule $\pi = (S, \overline{A})$ with n_a accepted jobs and a batch B_s for which $|B_s| > \left\lfloor \sqrt{\frac{4(1+k)(\alpha n_a + \beta(n_a-1))^{\frac{k}{k+1}}(\alpha n_a)^{\frac{1}{k+1}}\theta_{\max}^{n_a}}{(\alpha+\beta)\vartheta_{\min}}} + 1 \right\rfloor$. Clearly, $|B_s| \geq 2$. Let $l = \frac{|B_s|}{2}$ if $|B_s|$ is even, and $l = \frac{|B_s|-1}{2}$ otherwise. Now consider a new schedule $\pi' = (S', \overline{A})$ constructed by splitting batch B_s into two batches $\{J_{[l_{s-1}+l+2]}, J_{[l_{s-1}+l+3]}, \ldots, J_{[l_s]}\}$ and $\{J_{[l_{s-1}+1]}, J_{[l_{s-1}+2]}, \ldots, J_{[l_{s-1}+l+1]}\}$ without changing the job processing times. By Lemma 3.2, the optimal resource allocation $\mathbf{u}'^*$ for schedule π' is

$$u_i'^* = \begin{cases} u_i^*, & i = 1, \ldots, s, \\ \left(\frac{k\alpha(n_a-(l_s-l))}{\gamma_i}\right)^{\frac{1}{k+1}} \times \omega_i^{\frac{k}{k+1}}, & i = s+1, \\ \left(\frac{k\alpha(n_a-l_{i-2})}{\gamma_i}\right)^{\frac{1}{k+1}} \times \omega_i^{\frac{k}{k+1}}, & i = s+2, \ldots, m+1. \end{cases}$$

By equation (3.1), the difference between the objective values of schedules π' and π is

$$\begin{aligned} \Delta &= \alpha(n_a - l_{s-1}) \left(\left(\frac{\omega_s}{u_s^*}\right)^k + \sum_{j=l_{s-1}+l+1}^{l_s} \left(\frac{w_{[j]}}{v_{[j]}^*}\right)^k \right) \\ &\quad + \alpha(n_a - (l_s - l)) \left(\left(\frac{\omega_{s+1}}{u_{s+1}'^*}\right)^k + \sum_{j=l_{s-1}+1}^{l_{s-1}+l} \left(\frac{w_{[j]}}{u_{[j]}^*}\right)^k \right) \\ &\quad - \alpha(n_a - l_{s-1}) \left(\left(\frac{\omega_s}{u_s^*}\right)^k + \sum_{j=l_{s-1}+1}^{l_s} \left(\frac{w_{[j]}}{v_{[j]}^*}\right)^k \right) \\ &\quad - \beta l \sum_{j=l_{s-1}+l+1}^{l_s} \left(\frac{w_{[j]}}{v_{[j]}^*}\right)^k \\ &\quad + \alpha \sum_{j=s+1}^{m} (n_a - l_{j-1}) \left(\left(\frac{\omega_{j+1}}{u_{j+1}'^*}\right)^k - \left(\frac{\omega_j}{u_j^*}\right)^k \right) \\ &\quad + \gamma_{s+1} u_{s+1}'^* + \sum_{j=s+1}^{m} \gamma_j \left(u_{j+1}'^* - u_j'^*\right) \\ &= -\alpha(|B_s| - l) \sum_{j=l_{s-1}+1}^{l_{s-1}+l} \left(\frac{w_{[j]}}{v_{[j]}^*}\right)^k - \beta l \sum_{j=l_{s-1}+l+1}^{l_s} \left(\frac{w_{[j]}}{v_{[j]}^*}\right)^k \\ &\quad + \alpha(n_a - (l_s - l)) \left(\frac{\omega_{s+1}}{u_{s+1}'^*}\right)^k \\ &\quad + \alpha \sum_{j=s+1}^{m} (n_a - l_{j-1}) \left(\left(\frac{\omega_{j+1}}{u_{j+1}'^*}\right)^k - \left(\frac{\omega_j}{u_j^*}\right)^k \right) \\ &\quad + \gamma_{s+1} u_{s+1}'^* + \sum_{j=s+1}^{m} \gamma_j \left(u_{j+1}'^* - u_j'^*\right) \end{aligned}$$

$$
\begin{aligned}
&= -\alpha(|B_s|-l)\sum_{j=l_{s-1}+1}^{l_{s-1}+l}\frac{\vartheta_{[j]}}{(k(\alpha(n_a-l_{s-1})+\beta(j-1-l_{s-1})))^{\frac{k}{k+1}}}\\
&\quad-\beta l\sum_{j=l_{s-1}+l+1}^{l_s}\frac{\vartheta_{[j]}}{(k(\alpha(n_a-l_{s-1})+\beta(j-1-l_{s-1})))^{\frac{k}{k+1}}}\\
&\quad+k^{\frac{-k}{k+1}}\left(\alpha(n_a-(l_s-l))\right)^{\frac{1}{k+1}}\theta_{s+1}\\
&\quad+k^{\frac{-k}{k+1}}\sum_{j=s+1}^{m}\left(\alpha(n_a-l_{j-1})\right)^{\frac{1}{k+1}}(\theta_{j+1}-\theta_j)\\
&\quad+k^{\frac{1}{k+1}}\left(\alpha(n_a-(l_s-l))\right)^{\frac{1}{k+1}}\theta_{s+1}\\
&\quad+k^{\frac{1}{k+1}}\sum_{j=s+1}^{m}\left(\alpha(n_a-l_{j-1})\right)^{\frac{1}{k+1}}(\theta_{j+1}-\theta_j)\\
&= -\alpha(|B_s|-l)\sum_{j=l_{s-1}+1}^{l_{s-1}+l}\frac{\vartheta_{[j]}}{(k(\alpha(n_a-l_{i-1})+\beta(j-1-l_{i-1})))^{\frac{k}{k+1}}}\\
&\quad-\beta l\sum_{j=l_{s-1}+l+1}^{l_s}\frac{\vartheta_{[j]}}{(k(\alpha(n_a-l_{i-1})+\beta(j-1-l_{i-1})))^{\frac{k}{k+1}}}\\
&\quad+\overline{k}\Bigg(\left(\alpha(n_a-(l_s-l))\right)^{\frac{1}{k+1}}\theta_{s+1}\\
&\quad+\sum_{j=s+1}^{m}\left(\alpha(n_a-l_{j-1})\right)^{\frac{1}{k+1}}(\theta_{j+1}-\theta_j)\Bigg)\\
&\le -(|B_s|-l)l(\alpha+\beta)\frac{\vartheta_{\min}}{(k(\alpha n_a+\beta(n_a-1)))^{\frac{k}{k+1}}}\\
&\quad+\overline{k}\left(\alpha n_a\right)^{\frac{1}{k+1}}\left(\theta_{s+1}+\sum_{j=s+1}^{m}(\theta_{j+1}-\theta_j)\right)\\
&\le -(|B_s|-l)l(\alpha+\beta)\frac{\vartheta_{\min}}{(k(\alpha n_a+\beta(n_a-1)))^{\frac{k}{k+1}}}\\
&\quad+\overline{k}\left(\alpha n_a\right)^{\frac{1}{k+1}}\theta_{m+1}\\
&\le -(|B_s|-l)l(\alpha+\beta)\frac{\vartheta_{\min}}{(k(\alpha n_a+\beta(n_a-1)))^{\frac{k}{k+1}}}\\
&\quad+\overline{k}\left(\alpha n_a\right)^{\frac{1}{k+1}}\theta_{\max}^{n_a}.
\end{aligned}
$$

Now, if $|B_s|$ is even, then $(|B_s|-l)l=\frac{|B_s|^2}{4}$; otherwise, $(|B_s|-l)l=\frac{|B_s|^2-1}{4}$. Since $|B_s|>\left\lfloor\sqrt{\frac{4(1+k)(\alpha n_a+\beta(n_a-1))^{\frac{k}{k+1}}(\alpha n_a)^{\frac{1}{k+1}}\theta_{\max}^{n_a}}{(\alpha+\beta)\vartheta_{\min}}}+1\right\rfloor$, one have

$$
\begin{aligned}
(|B_s|-l)l(\alpha+\beta)\frac{\vartheta_{\min}}{(k(\alpha n_a+\beta(n_a-1)))^{\frac{k}{k+1}}}&\ge\frac{|B_s|^2-1}{4}\times\frac{(\alpha+\beta)\vartheta_{\min}}{(k(\alpha n_a+\beta(n_a-1)))^{\frac{k}{k+1}}}\\
&>\overline{k}\left(\alpha n_a\right)^{\frac{1}{k+1}}\theta_{\max}^{n_a},
\end{aligned}
$$

which implies that $\Delta<0$. Therefore, π' is a better schedule than π, as required. □

Lemma 3.6. *For the problem* $1|covx, rej, \boldsymbol{B}|\sum_{j=1}^{n_a}(\alpha D_{[j]}+\beta H_{[j]}+\delta_{[j]}v_{[j]})+\sum_{i=1}^{m}\gamma_i u_i+\sum_{j\in\overline{A}}e_j$, *an optimal schedule exists in which the jobs in any two consecutive batches* B_s *and* B_{s+1} *satisfy* $\vartheta_i \le \vartheta_j$ *if* J_i *is in* B_s *and* J_j *is in* B_{s+1}.

Proof. Because of Lemma 3.2, the condition of Lemma 3.6 holds for an optimal schedule if and only if $\vartheta_i \le \vartheta_j$, where J_i is the first job in B_s and J_j is the last job in B_{s+1}. Now suppose that there exists an optimal schedule $\pi = (S, \overline{A})$ with n_a accepted jobs and two consecutive batches B_s and B_{s+1} in which $\vartheta_i > \vartheta_j$, where J_i is the first job in B_s and J_j is the last job in B_{s+1}. Consider a new schedule $\pi' = (S', \overline{A})$ constructed by swapping jobs J_i and J_j without changing the setup and job processing times. By equation (3.4), the difference between the objective values of schedules π' and π is

$$\begin{aligned}\Delta &= -\overline{k}(\alpha(n_a - l_{s-1}))^{\frac{1}{k+1}}(\vartheta_i - \vartheta_j) + \overline{k}(\alpha(n_a - l_s) + \beta(|B_{s+1}| - 1))^{\frac{1}{k+1}}(\vartheta_i - \vartheta_j)\\ &= \overline{k}\Big((\alpha(n_a - l_s) + \beta(|B_{s+1}| - 1))^{\frac{1}{k+1}} - (\alpha(n_a - l_{s-1}))^{\frac{1}{k+1}}\Big)(\vartheta_i - \vartheta_j)\\ &= \overline{k}\Big((\beta(n_a - l_{s-1}) - \alpha|B_s| + \beta(|B_{s+1}| - 1))^{\frac{1}{k+1}} - (\alpha(n_a - l_{s-1}))^{\frac{1}{k+1}}\Big)(\vartheta_i - \vartheta_j),\end{aligned}$$

which is negative because $\alpha \ge \beta$, $|B_s| > |B_{s+1}| - 1$, and $\vartheta_i > \vartheta_j$, implying that schedule π' is at least as good as π. Repeating this modifying argument for all the accepted jobs not sequenced in the specified order yields the result. □

In what follows, let $(i, j, x, y, r)_{n_a}, F_{n_a}(i, j, x, y, r), \phi_{n_a}(i, j, x, y, r)$, and $\psi_{n_a}(i, j, x, y, r)$ be defined the same as in Section 2, and suppose that the jobs are indexed in a nondecreasing order of ϑ_j. Based on the results developed above, a recursion relation is next provided that can be exploited to design a polynomial-time dynamic programming solution algorithm with a modification of Algorithm 1 for the problem $1|covx, rej, \mathbf{B}|\sum_{j=1}^{n_a}(\alpha D_{[j]} + \beta H_{[j]} + \delta_{[j]}v_{[j]}) + \sum_{i=1}^{m}\gamma_i u_i + \sum_{j\in\overline{A}} e_j$ as follows:

Algorithm 2.

Denote $\widetilde{B}_{n_a} = \left\lfloor \sqrt{\frac{4(1+k)(\alpha n_a+\beta(n_a-1))^{\frac{k}{k+1}}(\alpha n_a)^{\frac{1}{k+1}}\theta_{\max}^{n_a}}{(\alpha+\beta)\vartheta_{\min}}} + 1 \right\rfloor$ for $n_a = 0, 1, \ldots, n$, and calculate $\overline{k}$ and ϑ_j according to equations (3.5) and (3.7) for $j = 1, \ldots, n$. Re-number the jobs such that $\vartheta_1 \le \vartheta_2 \le \ldots \le \vartheta_n$.

(1) Initial conditions

(a) If (n_a, i, j, x, y, r) does not satisfy $0 \le n_a \le n$, $1 \le i \le n$, $0 \le j \le \min\{i, n_a\}$, $r = j = 0$ or $1 \le r \le j$, $0 \le x \le \min\{n_a, \widetilde{B}_{n_a}\}$, and $0 \le y \le \min\{j, x\}$, then

$$F_{n_a}(i, j, x, y, r) = +\infty, \phi_{n_a}(i, j, x, y, r) = \emptyset, \psi_{n_a}(i, j, x, y, r) = \emptyset.$$

(b) If $x = r = 0$ and $n_a \ge 0$, then

$$\begin{aligned}&F_{n_a}(1, 0, 0, 0, 0) = e_1,\\ &\phi_{n_a}(1, 0, 0, 0, 0) = \phi,\ \psi_{n_a}(1, 0, 0, 0, 0) = 0.\end{aligned}$$

(c) If $1 \le x \le \min\{n_a, \widetilde{B}_{n_a}\}$ and $n_a \ge 1$, then

$$\begin{aligned}&F_{n_a}(1, 1, x, 1, 1) = \overline{k}\left(\theta_1(\alpha n_a)^{\frac{1}{k+1}} + \vartheta_1(\alpha n_a + \beta(x-1))^{\frac{1}{k+1}}\right),\\ &\phi_{n_a}(1, 1, x, 1, 1) = x,\ \psi_{n_a}(1, 1, x, 1, 1) = 1.\end{aligned}$$

(2) Recursive relations

For $0 \le n_a \le n$, $2 \le i \le n$, $0 \le j \le \min\{i, n_a\}$, $r = j = 0$ or $1 \le r \le j$, and $0 \le x \le \min\{n_a, \widetilde{B}_{n_a}\}$:

(a) For $2 \le y \le \min\{j, x\}$:

$$F_{n_a}(i,j,x,y,r) = \min \begin{cases} F_{n_a}(i-1,j,x,y,r) + e_i, \ (\equiv X) \\ F_{n_a}(i-1,j-1,x,y-1,r) \\ \quad +\overline{k}\vartheta_i \left(\alpha(n_a - j + y) + \beta(x-y)\right)^{\frac{1}{k+1}}, \ (\equiv Y) \end{cases}$$

$$\phi_{n_a}(i,j,x,y,r) = \begin{cases} \phi, & \text{if } F_{n_a}(i,j,x,y,r) = X, \\ j + x - 2y + 1, & \text{if } F_{n_a}(i,j,x,y,r) = Y, \end{cases}$$

$$\psi_{n_a}(i,j,x,y,r) = 0.$$

(b) For $y = 1$:

$$F_{n_a}(i,j,x,1,r) = \min \begin{cases} F_{n_a}(i-1,j,x,y,r) + e_i, \ (\equiv U) \\ \min_{z \ge x}\{F_{n_a}(i-1,j-1,z,z,r-1)\} \\ \quad +\overline{k}\big(\theta_r \left(\alpha(n_a - j + 1)\right)^{\frac{1}{k+1}} \\ \quad +\vartheta_i \left(\alpha(n_a - j + 1) + \beta(x-1)\right)^{\frac{1}{k+1}}\Big), \ (\equiv V) \end{cases}$$

$$\phi_{n_a}(i,j,x,1,r) = \begin{cases} \phi, & \text{if } F_{n_a}(i,j,x,1,r) = U, \\ j + x - 1, & \text{if } F_{n_a}(i,j,x,1,r) = V, \end{cases}$$

$$\psi_{n_a}(i,j,x,1,r) = \begin{cases} 0, & \text{if } F_{n_a}(i,j,x,1,r) = U, \\ 1, & \text{if } F_{n_a}(i,j,x,1,r) = V. \end{cases}$$

(3) The optimal solution value is $F^* = \min\{F_{n_a}(n, n_a, x, x, r) | 0 \le n_a \le n, 0 \le x \le \min\{n_a, \widetilde{B}_{n_a}\}, r = n_a = 0 \text{ or } 1 \le r \le n_a\}$.

Let n_a^*, x^*, and r^* be the corresponding values. The optimal sequence can be obtained by recursively searching the position functions $\phi_{n_a^*}(i,j,x,y,r)$ beginning with $\phi_{n_a^*}(n, n_a^*, x^*, x^*, r^*)$. The optimal batch delivery date of each job can be determined from the batch indicator functions $\psi_{n_a^*}(i,j,x,y,r)$ and the optimal resource allocation $(\mathbf{u}^*, \mathbf{v}^*)$ can be calculated by equations (3.2) and (3.3).

Theorem 3.7. *The problem* $1|covx, rej, \boldsymbol{B}|\sum_{j=1}^{n_a}(\alpha D_{[j]} + \beta H_{[j]} + \delta_{[j]} v_{[j]}) + \sum_{i=1}^{m} \gamma_i u_i + \sum_{j \in \overline{A}} e_j$ *can be solved in* $O(n^6)$ *time by Algorithm* 2.

Proof. The proof is similar to that of Theorem 2.5. □

Note that for the case where $\theta_i = \theta$ for $i = 1, \ldots, m$, which implies that the setup is a constant, one can drop the decision variable r in Algorithm 2 and obtain the following result.

Theorem 3.8. *The problem* $1|covx, rej, \boldsymbol{B}|\sum_{j=1}^{n_a}(\alpha D_{[j]} + \beta H_{[j]} + \delta_{[j]} v_{[j]}) + \sum_{i=1}^{m} \gamma_i u_i + \sum_{j \in \overline{A}} e_j$ *can be solved in* $O(n^5)$ *time in the case with a common setup time.*

4. Conclusions

This paper considers single-machine batch scheduling with job reject penalty, batch availability, and batch-dependent setup times, under the assumption that both the setup and job processing times are convex decreasing functions of the amounts of a resource allocated to the operations. This paper aims to solve this more realistic and complex scheduling model. Specially, it is shown that both the problems $1|rej, \mathbf{B}|\sum_{j=1}^{n_a}(\alpha D_{[j]}+\beta H_{[j]})+\sum_{j\in\overline{A}} e_j$ and $1|covx, rej, \mathbf{B}|\sum_{j=1}^{n_a}(\alpha D_{[j]}+\beta H_{[j]}+\delta_{[j]}v_{[j]})+\sum_{i=1}^{m}\gamma_i u_i+\sum_{j\in\overline{A}} e_j$ can be solved in $O(n^6)$ by dynamic programming algorithms. It is also shown that the case of both problems with a common setup time can be solved in $O(n^5)$ time.

Several important issues are interesting for future research.

- Ascertaining the computational complexity status of the case of the problem where $\alpha < \beta$ since in this case Lemmas 2.3 and 3.6 may not hold;
- Investigating the case that the processing times (setup times) can not be less than a minimum duration;
- Extending our model to consider the linear resource consumption functions;
- Extending our model to different machine environments (*e.g.*, flowshop).

Acknowledgements. We are grateful to the Editor-in-Chief and two anonymous referees for their many helpful comments on earlier versions of our paper. This work was supported in part by the Ministry of Science Technology (MOST) of Taiwan under grant numbers MOST 105-2221-E-035-053-MY3 and MOST 103-2410-H-035-022-MY2.

References

[1] A. Allahverdi, J.N.D. Gupta and T. Aldowaisan, A review of scheduling research involving setup considerations. *Omega* **27** (1999) 219–239.

[2] Y. Bartal, S. Leonardi, A. Marchetti-Spaccamela, J. Sgall and L. Stougie, Multiprocessor scheduling with rejection. in: *Seventh ACM-SIAM Symp. Discrete Algorithms* (2000) 95–103.

[3] T.C.E. Cheng and M.Y. Kovalyov, Single machine batch scheduling with deadlines and resource dependent processing times. *Oper. Res. Lett.* **17** (1995) 243–249.

[4] T.C.E. Cheng, A. Janiak and M.Y. Kovalyov, Single machine batch scheduling with resource dependent setup and processing times.*Eur. J. Oper. Res.* **135** (2001) 177–183.

[5] G.H. Hardy, J.E. Littlewood and G. Polya, Inequalities. Cambridge University Press, Cambridge (1934).

[6] C.L. Monma, A. Schrijver, M.J. Todd and V.K. Wei, Convex resource allocation problems on directed acyclic graphs: Duality, complexity, special cases and extensions. *Math. Oper. Res.* **15** (1990) 736–748.

[7] G. Mosheiov and D. Oron, Single machine scheduling with batch-dependent setup times. *Inf. Proc. Lett.* **98** (2006) 73–78.

[8] E. Nowicki and S. Zdrzalka, A survey of results for sequencing problems with controllable processing times. *Discrete Appl. Math.* **26** (1990) 271–287.

[9] C.N. Potts and M.Y. Kovalyov, Scheduling with batching: A review. *Eur. J. Oper. Res.* **120** (2000) 228–249.

[10] C.N. Potts and L.N. Van Wassenhove, Integrating scheduling with batching and lot-sizing: a review of algorithms and complexity. *J. Oper. Res. Soc.* **46** (1991) 395–406.

[11] D. Shabtay, The single machine serial batch scheduling problem with rejection to minimize total completion time and total rejection cost. *Eur. J. Oper. Res.* **233** (2014) 64–74.

[12] D. Shabtay, N. Gaspar and L. Yedidsion, A bicriteria approach to scheduling a single machine with job rejection and positional penalties. *J. Comb. Optim.* **23** (2012) 395–424.

[13] D. Shabtay and G. Steiner, A survey of scheduling with controllable processing times. *Discrete Appl. Math.* **155** (2007a) 1643–1666.

[14] D. Shabtay and G. Steiner, Single machine batch scheduling to minimize total completion time and resource consumption costs. *J. Scheduling* **10** (2007b) 255–261.

[15] R.G. Vickson, Two single-machine sequencing problems involving controllable job processing times. *AIIE transactions* **12** (1980a) 258–262.

[16] R.G. Vickson, Choosing the job sequence and processing times to minimize total processing plus flow cost on a single machine. *Oper. Res.* **29** (1980b) 1155–1167.

[17] Y. Yin, T.C.E. Cheng, C.-J. Hsu and C.-C. Wu, Single-machine batch delivery scheduling with an assignable common due window. *Omega* **41** (2013b) 216–225.

[18] Y. Yin, T.C.E. Cheng, D, Wang and C.-C. Wu, Improved algorithms for single-machine serial-batch scheduling with rejection to minimize total completion time and total tejection cost. *IEEE Trans. Syst. Man Cybernetics-Systems* **46** (2016) 1578–1588.

[19] Y. Yin, S.-R. Cheng, C.-C. Wu and S.-R. Cheng, Single-machine common due-date scheduling with batch delivery costs and resource-dependent processing times. *Int. J. Production Res.* **51** (2013a) 5083–5099.

[20] Y. Yin, D. Ye, and G. Zhang, Single machine batch scheduling to minimize the sum of total flow time and batch delivery cost with an unavailability interval. *Information Sci.* **274** (2014) 310–322.