A unified approach for single machine scheduling with position-dependent workloads and positional penalties

Baruch Mor¹

Received: 1 August 2019 / Accepted: 3 January 2020 / Published online: 14 January 2020
© Springer Nature Switzerland AG 2020

Abstract

In this paper, we provide unified methods for solving scheduling problems with *convex resource allocation*, concentrating on objective functions that can be expressed as a scalar product of the actual processing times vector and a job-independent and position-dependent penalties vector. We assume *general position-dependent workloads*, i.e., the workloads are not restricted to be either monotone functions of the job positions or to specific functions. The first unified method focuses on minimizing a scheduling measure subject to a constraint on the resource consumption, whereas the second addresses the complementary problem of minimizing the consumed resource given an upper bound on the scheduling measure. For each unified approach, we provide a methodical analysis and consequently provide a $O(n^3)$ solution algorithm. Furthermore, to demonstrate the unified schemes, we solve several scheduling measures that involve earliness and tardiness penalties.

Keywords Single machine scheduling · Convex resource allocation · Position-dependent workloads · Positional penalties

1 Introduction

Classical deterministic scheduling theory assumed that the jobs processing times are given constants. Later studies challenged this paradigm and introduced position-dependent processing times or controllable processing times. Studies involving position-dependent processing times are derived from the notion that processing times can be decreased due to a learning effect or increased implying an aging effect or deteriorating jobs. Generally, the optimal job sequence can be achieved by a simple $O(n \log n)$ matching procedure, i.e., matching n jobs to n positions (see [9]). The most inclusive representation of the variable job processing times phenomenon presumes general position-dependent processing times, and accordingly no restrictions on the jobs functional form are imposed. In these scheduling problems, the input includes a matrix of $n \times n$ job-position processing

times. Consequently, the computational complexity of the solution algorithm increases significantly, as in most cases the solution requires solving a Linear Assignment Problem (LAP), which is known to be solved in $O(n^3)$ time. Widespread research has been carried out addressing this phenomenon. Recent studies on various variable job processing time problems include among others: [2, 4–6, 17, 19, 20, 36, 38, 39, 40, 41].

Scheduling models with controllable processing times are very extensive and influential. Similar to variable processing times research, studies with controllable processing times adopt the principle that processing times are not fixed. Unlike the job-position method, in this scheme the processing times are not a function of the position/time, but rather a function of the resources allocated to the job operations. This modernized methodology aims to address various real-life problems where the improvement in the deployment of inputs in the production system is

✉ Baruch Mor, baruchm@ariel.ac.il | ¹Department of Economics and Business Administration, Ariel University, 40700 Ariel, Israel.

realized by allocating extended resource to job processing. Most studies suppose that the actual job processing time is either a linear or non-linear function of the amount of resource allocated to the processing of the job. In this study, we focus on the latter and more practical function. Thus, we assume that the input includes position-dependent workloads of the jobs processing operation. In fact, the actual job processing times is a convex decreasing function of the amount of resource allocated to the processing of the job. According to a wide-ranging survey of Shabtay and Steiner [25], the convex function reflects the fundamental law of diminishing marginal returns, validating that productivity increases at a decreasing rate with the utilized amount of resource. Moreover, [25] provides numerous practical problems involving various objective functions on different machine settings varying from single machine to open-shops, demonstrating the importance of this research class in scheduling theory. For instance, [25] points out that convex resource allocation can be found in “many actual government and industrial operations and ... very large scale integration (VLSI) circuit design, where the product of the silicon area (resource) and the square of time spent equals a constant value (the workload) for an individual job”. Recent articles with convex resource allocation include [6, 7, 14, 18, 19, 26–30, 34, 41].

For a complete exposition, we note that [25] summarized that there are four different types of scheduling problems with controllable processing times:

Type 1: Minimizing the total integrated cost of the scheduling measure and consumed resource.

Type 2: Minimizing the scheduling measure subject to an upper bound on the available resource.

Type 3: Minimizing the consumed resource subject to an upper bound on the scheduling measure.

Type 4: Identifying the set of Pareto-optimal schedules for the scheduling measure and the consumed resource.

Recently, Oron [24] combined the methods of position-dependent processing times and convex resource allocation by introducing the realistic and challenging model of *position-dependent workloads*. The author assumed a continuous non-renewable resource, implying that any amount of available resource can be allocated to any given job. Specifically, he considered decreasing workloads as a result of a position-dependent *learning effect*. The paper focused on Type 2 problems, i.e., minimizing different scheduling measures subject to a given upper bound on the available resource. The studied scheduling measures were: makespan; total completion time (flowtime); linear combination of the makespan and total flowtime on a single machine; and minimizing total flowtime on m -identical

parallel machines. Moreover, the author suggested that minimizing Total Absolute Deviation of Completion times (TADC) and minimizing Total Absolute Deviation of Waiting Times (TADW) can be solved using the same approach.

Leyvand et al. [13] provided a unified mechanism for scheduling with convex resource consumption functions using positional penalties, and focused on Type 1 problems. The authors proved that scheduling problems with *position-independent* workloads, where the objective functions are the scalar products of two vectors, i.e., the (actual) processing times vector and the positional weights vector, can be solved in polynomial time by applying a unified method.

In this paper, we further generalize the results of [13] and [24]. We consider scheduling problems involving *general position-dependent* and *positional penalties* and make available a unified tool for solving such problems. We prove that the results of [24] can be further extended to solve problems where the workloads are not restricted to monotone functions of the job starting times, functions of the job positions, or to specific linear/exponential functions.

Following [13] and [24], we restrict our study to objective functions that can be expressed as a function of the jobs processing times and the positional weights. Unlike [13], who provided a unified method for the Type 1 problems and [24] who studied Type 2 problems, we provide unified methods both for Types 2 and 3.

In the first part of this study, we provide a unified scheme for minimizing a general scheduling measure that is a scalar product of the actual processing times and the positional weights vectors, subject to a constraint on the available amount of resource that can be consumed. Similar to [24], we prove that these problems can be solved in $O(n^3)$ time. Once we obtain the solution algorithm for the general problem, we apply it with minor modifications to solve three different scheduling problems with earliness and tardiness penalties. Specifically, we consider the following objective functions: the total weighted earliness plus the total weighted tardiness with common unrestrictive due-date; minsum common due-window; and minsum common due-window based on flow-allowance. In the second part, we introduce a unified scheme for minimizing the consumed resource subject to an upper bound on the scheduling measure and apply the results to solve the same earliness/tardiness problems.

The article is comprised of five major sections and several sub-sections. In Sect. 2, we provide the notations and formulation of all studied problems. Section 3 is dedicated to minimizing scheduling measures: In Sect. 3.1, we introduce the unified method and in subsequent Sects. 3.2–3.4, we solve total weighted earliness and tardiness with common unrestrictive due-date, minsum common

due-window, and minsum common due-window based on flow-allowance, respectively. In Sect. 4, we concentrate our attention on the unified method for minimizing the total amount consumed. In Sect. 5, we provide conclusions and future research topics.

2 Notations and formulation

We consider an n -job single machine scheduling problem. We assume, as is very prevalent in scheduling theory, that all the jobs are available at time zero, and preemption is forbidden. The actual processing time p_{jr} of job $j, j = 1, \dots, n$ if assigned to position $r, r = 1, \dots, n$ is defined as $p_{jr} = \left(\frac{w_{jr}}{u_j}\right)^k$, where w_{jr} is a strictly positive value emulating the workload of the processing operation of job j in position r , u_j is a decision variable reflecting the amount of resource assigned to the processing operation, and k is a given positive factor.

During the course of this study, we consider scheduling problems with cost functions that can be expressed as the scalar product of the actual processing times and the positional weights vectors. The positional weights are denoted by $\xi_r, r = 1, \dots, n$. The vector of decision variables $u_j, j = 1, \dots, n$ is denoted by $\mathbf{u}$. We also denote by X_{jr} the set binary decision variables such that,

$$X_{jr} = \begin{cases} 1, & \text{if job } j \text{ is assigned to position } r \\ 0, & \text{otherwise} \end{cases}, j, r = 1, \dots, n.$$

Consequently, the general cost function can be formulated as,

$$Z_0(\mathbf{u}) = \sum_{j=1}^n \sum_{r=1}^n \xi_r p_{jr} X_{jr} = \sum_{j=1}^n \sum_{r=1}^n \xi_r \left(\frac{w_{jr}}{u_j}\right)^k X_{jr}$$

In the first part, we seek to minimize cost functions of this form subject to a single constraint, i.e., that the total amount of resource allocated to all jobs cannot exceed a given upper bound denoted by U . Using the conventional triplet notation, proposed by [8], the general problem studied here, denoted by **PZ0**, is:

$$\mathbf{PZ0} : 1 \left| \sum_{j=1}^n u_j \leq U \right| Z_0.$$

After we solve this general problem, we apply the devised solution procedure to solve three different scheduling problems involving positional weights. To formulate the additional problems, we need several new notations as follows.

For a given schedule, the completion time of job $j, j = 1, \dots, n$ is denoted by C_j . Let, d_j denote the due-date of job $j, j = 1, \dots, n$. Jobs completed before the due-date are considered as early, and jobs completed after the due-date are considered as tardy. The (single) on-time job is not penalized whereas the early and tardy jobs incur penalties. The earliness of the job j is defined as $E_j = \max\{0, d_j - C_j\}$ and the tardiness of job j is defined as $T_j = \max\{0, C_j - d_j\}, j = 1, \dots, n$.

We first apply the unified approach to minimize the total weighted earliness plus the total weighted tardiness with a common unrestrictive due-date (URDD). We denote the common due-date by d , such that $d_j = d, j = 1, \dots, n$. We assume that the earliness penalty and the tardiness penalty are strictly positive and not necessarily equal. Let α denote the earliness unit penalty and β denote the tardiness unit penalty, thus the objective function is, $Z_1 = \sum_{j=1}^n \alpha E_j + \sum_{j=1}^n \beta T_j$.

As mentioned, we seek to minimize the objective function subject to a constraint on the total consumed resource, thus the first application of our study is the problem:

$$\mathbf{PZ1} : 1 \left| \text{URDD}, \sum_{j=1}^n u_j \leq U \right| Z_1.$$

Next, we consider the very influential common due-window (CONDW) assignment and scheduling problem first introduced by [15]. Let d_1 and d_2 denote the starting time and completion time of the due-window, respectively. We also denote the due-window size by $D = d_2 - d_1$, such that $d_2 \geq d_1$. Unlike the previous problem where the common due-date is a part of the input, in the current problem the due-dates are decision variables (in conjunction with the amount of resource allocated to each job). To produce solutions with early and reduced due-window size, there are four cost components: earliness, tardiness, cost of delaying, and widening the due-window. The earliness of j is $E_j = \max\{0, d_1 - C_j\}$ and the tardiness of job j is $T_j = \max\{0, C_j - d_2\}, j = 1, \dots, n$. Thus, jobs that are completed within the due-window are not penalized, whereas jobs that are completed prior to or after the window are penalized according to their earliness or tardiness. Let α denote the earliness unit penalty, β denote the tardiness unit penalty, γ denote the unit penalty of delaying the due-window, and δ denote the unit penalty of extending the due-window. The objective function is given by $Z_2 = \sum_{j=1}^n \alpha E_j + \sum_{j=1}^n \beta T_j + \gamma d_1 + \delta D$. Thus, we seek simultaneously the optimal job sequence and the due-window parameters that minimize the scheduling measure, implying that the third problem is:

$$\mathbf{PZ2} : 1 \left| \text{CONDW}, \sum_{j=1}^n u_j \leq U \right| Z_2.$$

The last problem we solve, using the unified approach, addresses the due-window assignment and scheduling problem based on common flow-allowance (SLKDW), presented by [23]. Similar to problem **PZ1**, jobs that are completed within the time interval (due-window) are not penalized and otherwise are penalized according to their earliness/tardiness. Once more, we seek instantaneously the optimal job sequence and the start time and size of the window. In this method of due-window problems, there are two common flow-allowance constants denoted by q_1 and q_2 , which are decision variables, such that $q^2 \geq q^1$. Unlike problem **PZ1**, the due-windows, denoted by D_j , are job-dependent such that $D_j = [d_{1j}, d_{2j}]$, $j = 1, \dots, n$, where $d_{ij} = p_j + q_i$, $i = 1, 2$, $j = 1, \dots, n$.

We note that since the flow-allowance parameters are job-independent, the size of the due-window is identical for all jobs $D_j = d_{2j} - d_{1j} = p_j + q_2 - (p_j + q_1) = q_2 - q_1 = D$, $j = 1, \dots, n$. The earliness and tardiness of j is $E_j = \max\{0, d_{1j} - C_j\}$ and $T_j = \max\{0, C_j - d_{2j}\}$, $j = 1, \dots, n$, respectively. Our aim is to find the early and narrow interval size possible for all jobs. There are four cost components: earliness, tardiness, costs of delaying the due-window, and extending the due-window size. Let α denote the unit earliness cost, β denote the unit tardiness cost, γ denote the unit cost of q_1 , and δ denote the unit cost of lengthening $D = q_2 - q_1$. Hence, the scheduling measure in this case is $Z_3 = \sum_{j=1}^n \alpha E_j + \sum_{j=1}^n \beta T_j + n\gamma q_1 + n\delta D$ and the last problem studied in this section is:

$$\mathbf{PZ3} : 1 \left| \text{SLKDW}, \sum_{j=1}^n u_j \leq U \right| Z_3.$$

In the second part of the paper, we address the complementary problem of **PZ0**. Now we provide a unified approach for solving problems where the goal is to minimize the total amount of consumed resource, U , subject to a constraint on the scheduling measure, denoted by Q . Thus, the general problem is:

$$\mathbf{PU0} : 1 | Z_0 \leq Q | U.$$

Where, as throughout this study, the scheduling measure is a function of the processing times and positional weights:

$$Z_0(\mathbf{u}) = \sum_{j=1}^n \sum_{r=1}^n \xi_r p_{jr} X_{jr} = \sum_{j=1}^n \sum_{r=1}^n \xi_r \left(\frac{w_{jr}}{u_j} \right)^k X_{jr}.$$

Again, the main goal of this part is to apply the results achieved in the study of the general case to solve specific scheduling problems. Therefore, we address minimizing the total consumed resource with the same scheduling measures considered in the first part. Formally, the problems that we solve in this part are:

$$\mathbf{PU1} : 1 | \text{URDD}, Z_1 \leq Q | U.$$

$$\mathbf{PU2} : 1 | \text{CONDW}, Z_2 \leq Q | U.$$

$$\mathbf{PU3} : 1 | \text{SLKDW}, Z_3 \leq Q | U.$$

3 A unified approach for minimizing a scheduling measure

In this section, we establish the theoretical fundamentals of our unified approach to solving scheduling problems involving positional weights. In all problems, we seek to minimize a cost function subject to a constraint that the total amount of consumed resource cannot exceed a given upper bound. In Sub-Sect. 3.1, we analyze the theoretical problem and introduce a general solution procedure based on solving a Linear Assignment Problem (LAP). In the succeeding Sects. 3.2–3.4, we solve practical problems founded on the general procedure with very minor adaptations, while keeping the computational complexity.

3.1 Problem PZ0: $1 \left| \sum_{j=1}^n u_j \leq U \right| Z_0$

As mentioned in Sect. 2, the theoretical objective function that we study in this section, in its most general form, can be expressed as follows,

$$Z_0(\varphi, \mathbf{u}) = \sum_{j=1}^n \sum_{r=1}^n \xi_r p_{jr} X_{jr} = \sum_{j=1}^n \sum_{r=1}^n \xi_r \left(\frac{w_{jr}}{u_j} \right)^k X_{jr}.$$

We need to minimize the objective, subject to a constraint on the available resource. Thus, problem **PZ0** falls into the category of constrained Non-Linear Programming (NLP):

NLP 3.1.1

min Z_0

s.t.

$$\sum_{j=1}^n u_j \leq U$$

$$u_j \geq 0, j = 1, \dots, n.$$

Next, we utilize the Karush–Kuhn–Tucker (KKT) conditions to achieve an optimal solution (see [10]).

Theorem 3.1.1 Assume that $Z(\mathbf{u})$ and $\theta(\mathbf{u})$ are differentiable functions satisfying certain regularity conditions. Then $\mathbf{u}^* = (u_1^*, u_2^*, \dots, u_n^*)$ can be an optimal solution for the nonlinear programming problem only if there exists a number λ such that all KKT conditions are satisfied.

We emphasize that the objective function and the single constraint are convex functions of the amount of resource allocated to jobs $j = 1, \dots, n$, and the problem is to find the global minimum of a convex function under linear constraint. Thus, from Theorem 3.1.1, NLP 3.1.1 can be solved to optimality by using the KKT conditions.

In order to formulate the KKT conditions, we define the Lagrangian

$$\mathcal{L}_1(\varphi, \mathbf{u}, \lambda) = Z_0(\varphi, \mathbf{u}) + \lambda(\theta(\varphi, \mathbf{u}) - U),$$

where:

- i. $\mathbf{u} = (u_1, u_2, \dots, u_n)$ is the vector of multiple decision variables;
- ii. λ is the single Lagrange multiplier (the shadow price of the objective function);
- iii. $\theta(\varphi, \mathbf{u}) = \sum_{j=1}^n u_j$.

Subsequently, applying Theorem 3.1.1 yields the following KKT conditions for Problem **PZ0**:

1. $\frac{\partial Z_0(\varphi, \mathbf{u})}{\partial u_j} + \lambda \frac{\partial \theta(\varphi, \mathbf{u})}{\partial u_j} \geq 0$
2. $u_j \left(\frac{\partial Z_0(\varphi, \mathbf{u})}{\partial u_j} + \lambda \frac{\partial \theta(\varphi, \mathbf{u})}{\partial u_j} \right) = 0$
3. $\theta(\varphi, \mathbf{u}) - U \leq 0$
4. $\lambda(\theta(\varphi, \mathbf{u}) - U) = 0$
5. $u_j \geq 0, j = 1, \dots, n$
6. $\lambda \geq 0$

We start our discussion by assuming that the job sequence, φ , has been achieved already and seek the expressions for the optimal amount of resource allocated to each job and the optimal actual processing time. Once these equations are realized, we reinsert them to the cost function to establish the optimal cost.

In the given sequence, if job h is assigned to position l , then $X_{hl} = 1$, implying that differentiating the Lagrangian with respect to u_h yields:

$$\frac{\partial \mathcal{L}_1(\varphi, \mathbf{u}, \lambda)}{\partial u_h} = \frac{\partial Z_0(\varphi, \mathbf{u})}{\partial u_h} + \lambda \frac{\partial \theta(\varphi, \mathbf{u})}{\partial u_h} = -k \xi_l \frac{1}{u_h} \left(\frac{w_{hl}}{u_h} \right)^k + \lambda. \quad (1)$$

To satisfy Condition 2, we assume and prove later that $u_j > 0, j = 1, \dots, n$, thus from condition 2:

$$-k \xi_l \frac{1}{u_h} \left(\frac{w_{hl}}{u_h} \right)^k + \lambda = 0. \quad (2)$$

From (2) we obtain an equation for the Lagrange multiplier,

$$\lambda = k \xi_l \frac{1}{u_h} \left(\frac{w_{hl}}{u_h} \right)^k > 0. \quad (3)$$

Equation (1) is valid for each job j if assigned to position r in φ , i.e., if $X_{jr} = 1$, thus,

$$\lambda = k \xi_r \frac{1}{u_j} \left(\frac{w_{jr}}{u_j} \right)^k, j = 1, \dots, n, r = 1, \dots, n. \quad (4)$$

Thus, combining (3) and (4), we get the amount of resource allocated to job j if assigned to position r :

$$u_j = \left(\frac{\xi_r w_{jr}^k}{\xi_l w_{hl}^k} \right)^{\frac{1}{1+k}} u_{hr}, j = 1, \dots, n, r = 1, \dots, n. \quad (5)$$

From (3), $\lambda > 0$ and we conclude from Condition 4 that,

$$\sum_{j=1}^n u_j - U = 0. \quad (6)$$

Now that we have established the equation of amount of resource allocated to job $j, j = 1, \dots, n$, we return to the initial stage where the binary variables $X_{jr}, j, r = 1, \dots, n$ are still unknowns. Specifically, we combine Eqs. (5) and (6) to get an equation for the consumed resource as a function of the jobs workloads.

$$U = \sum_{j=1}^n u_j = \sum_{j=1}^n \sum_{r=1}^n u_j X_{jr} = u_h \left(\frac{1}{\xi_l w_{hl}^k} \right)^{\frac{1}{1+k}} \sum_{j=1}^n \sum_{r=1}^n \left(\xi_r w_{jr}^k \right)^{\frac{1}{1+k}} X_{jr}. \quad (7)$$

To ease the reading let,

$$W = \sum_{j=1}^n \sum_{r=1}^n \left(\xi_r w_{jr}^k \right)^{\frac{1}{1+k}} X_{jr} = \sum_{j=1}^n \sum_{r=1}^n \xi_r^{\frac{1}{1+k}} w_{jr}^{\frac{k}{1+k}} X_{jr}. \quad (8)$$

From (7), the amount of resource allocated to job j if assigned to position r , as a function of the input data is:

$$u_j = \frac{U}{W} \xi_r^{\frac{1}{1+k}} w_{jr}^{\frac{k}{1+k}} > 0, j = 1, \dots, n. \quad (9)$$

At this point, we note that all KKT conditions were met as follows:

- i. Conditions 1 and 2 are satisfied from (2).
- ii. Conditions 3 and 4 are satisfied from (6).
- iii. Condition 5 is satisfied from (9).

iv. Condition 6 is satisfied from (3).

Since all KKT conditions are met, we can use (9) to find an equation for the actual processing time of job j if assigned to position r ,

$$p_{jr} = \left(\frac{w_{jr}}{u_j} \right)^k = \left(\frac{W}{U} \right)^k \xi_r^{-\frac{k}{1+k}} w_{jr}^{\frac{k}{1+k}}, j, r = 1, \dots, n. \quad (10)$$

Finally, the cost function as a function of the input data is given by,

$$\begin{aligned} Z_0 &= \sum_{j=1}^n \sum_{r=1}^n \xi_r p_{jr} X_{jr} \\ &= \sum_{j=1}^n \sum_{r=1}^n \xi_r \left(\frac{W}{U} \right)^k \left(\xi_r^{-\frac{k}{1+k}} w_{jr}^{\frac{k}{1+k}} \right) X_{jr} \\ &= \left(\frac{1}{U} \right)^k \sum_{j=1}^n \sum_{r=1}^n W^k \xi_r^{\frac{1}{1+k}} w_{jr}^{\frac{k}{1+k}} X_{jr}. \end{aligned} \quad (11)$$

It remains to obtain the optimal sequence that minimizes Z_0 . To do this, it is sufficient to minimize the expression $\sum_{j=1}^n \sum_{r=1}^n \xi_r^{\frac{1}{1+k}} w_{jr}^{\frac{k}{1+k}} X_{jr}$, implying that we need to solve the following LAP, denoted LAP 3.1.1:

LAP 3.1.1

$$\min \sum_{j=1}^n \sum_{r=1}^n \xi_r^{\frac{1}{1+k}} w_{jr}^{\frac{k}{1+k}} X_{jr}$$

s.t.

$$\sum_{j=1}^n X_{jr} = 1, \quad r = 1, \dots, n$$

$$\sum_{r=1}^n X_{jr} = 1, \quad j = 1, \dots, n$$

$$X_{jr}, \text{ binary}, \quad j = 1, \dots, n, r = 1, \dots, n$$

where,

$$X_{jr} = \begin{cases} 1, & \text{if job } j \text{ is assigned to position } r \\ 0, & \text{otherwise} \end{cases}.$$

This result validates the outcome of [24], with the added advantage that the workloads are general position-dependent and are not restricted to any specific function.

Next, we further simplify the objective function expression, and note that once the optimal sequence is achieved, by combining (8) and (11) we get,

$$Z_0 = \frac{W^{1+k}}{U^k}. \quad (12)$$

The formal unified approach solution procedure consists of several easy-to-follow steps.

Algorithm 3.1.1

1. Determine the positional weights $\xi_r, r = 1, \dots, n$.
2. Apply LAP 3.1.1 to obtain an optimal sequence and the workloads $w_{jr}, j, r = 1, \dots, n$.
3. Use Eq. (8) to determine W .
4. Utilize Eq. (9) to calculate the amount of resource allocated to each job $u_j, j = 1, \dots, n$.
5. Apply Eq. (10) to compute the actual processing times $p_{jr}, j, r = 1, \dots, n$.
6. Use Eq. (12) to obtain the optimal cost function value Z_0^* .

Theorem 3.1.2 The computational complexity of Algorithm 3.1.1 is $O(n^3)$.

Proof Step 1: Is carried out in $O(n)$ time. Step 2: The computational complexity required for solving the assignment problem is known to be $O(n^3)$. Steps 3–6: Each of these steps is done at most in $O(n)$ time. Thus, the total complexity of the solution procedure is dominated by Step 2 and is $O(n^3)$.

We note that if the input is a vector of n position-independent workloads (as opposed to a matrix of $n \times n$ job-position workloads), applying LAP 3.1.1 is unnecessary, and we need to assign jobs to positions according to the HLP rule (see [9]), implying that the computational effort is reduced to $O(n \log n)$.

In the next subsections, we apply the unified approach to address three fundamental scheduling problems involving due-dates. Based on Algorithm 3.1.1, with minor adaptations, we provide $O(n^3)$ time polynomial solution procedures for all studied problems.

3.2 Problem PZ1: $1 \mid URDD, \sum_{j=1}^n u_j \leq U \mid Z_1$

The first scheduling measure we consider is the total weighted earliness plus the total weighted tardiness, where all jobs share a common unrestrictive due-date. In the context of this study, we define the unrestrictive due-date as sufficiently late so as not to affect the scheduling decisions. Baker and Scudder [3] provide several important properties of an optimal schedule which are still valid for our problem:

Property 3.2.1 There are no idle times between consecutive jobs.

Property 3.2.2 An optimal schedule exists in which $d = C_l$ for some l .

Property 3.2.3 The index of the on-time job is

$$l = \left\lceil \frac{n\beta}{\alpha + \beta} \right\rceil. \quad (13)$$

Property 3.2.3 implies that jobs in positions $1, \dots, l - 1$ are early and penalized as such, while the job in position l is on-time and thus not penalized, whereas jobs in positions $l + 1, \dots, n$ are tardy and penalized. Consequently, it is straightforward to define the positional weight of job j if assigned to position r .

Property 3.2.4 The positional weights are given by:

$$\xi_r = \begin{cases} \alpha(r - 1), & r = 1, \dots, l \\ \beta(n - r + 1), & r = l + 1, \dots, n \end{cases}. \quad (14)$$

We note that the positional weight of the job assigned to the first position is zero, thus no resource should be allocated to reduce the processing time of this job, leading to the following property:

Property 3.2.5 The total available resource should be allocated to the jobs in positions $2, \dots, n$.

Thus, once the optimal sequence was obtained, the implication of Property 3.2.4 is threefold:

$$1. \quad u_j = 0 \text{ if } X_{j1} = 1. \quad (15)$$

$$2. \quad p_{j1} = w_{j1} \text{ if } X_{j1} = 1. \quad (16)$$

3. Eqs. (8) and (9), given in Sect. 3.2, must be modified as follows,

$$W = \sum_{j=1}^n \sum_{r=2}^n \xi_r^{\frac{1}{1+k}} w_{jr}^{\frac{k}{1+k}} X_{jr}, \quad (17)$$

$$u_j = \frac{U}{W} \xi_r^{\frac{1}{1+k}} w_{jr}^{\frac{k}{1+k}}, j = 1, \dots, n, \quad r = 2, \dots, n. \quad (18)$$

Now, we are ready to provide the solution algorithm for problem **PZ1**:

Algorithm 3.2.1

1. Utilize Eqs. (13) and (14) to determine l and $\xi_r, r = 1, \dots, n$, respectively.
2. Apply LAP 3.1.1 to obtain an optimal sequence.
3. If $X_{j1} = 1$, set $u_j = 0$ and $p_{j1} = w_{j1}$.
4. Utilize Eq. (17) to calculate W .
5. Apply Eq. (18) to compute $u_j, j = 1, \dots, n, r = 2, \dots, n$.
6. Use Eq. (10) to determine $p_{jr}, j = 1, \dots, n, r = 2, \dots, n$.
7. Use Eq. (12) to obtain Z_1^* .

Numerical example 3.2.1 We solve the 8×8 job-position workloads problem given in Table 1.

The common (unrestrictive) due-date is $d = 400$, the cost parameters are $\alpha = 3$ and $\beta = 4$, the total amount of available resource is $U = 10$ and $k = 0.5$.

We apply Algorithm 3.1.1 to solve the problem. We note that the index of the on-time jobs is $l = 5$, implying that the jobs in positions $1, \dots, 4$ are early and the jobs in positions $6, 7$ and 8 are tardy. The positional weights, optimal sequence, jobs workloads, optimal amount of allocated resource to each job, optimal actual processing times, and cost are presented in Table 2.

3.3 Problem PZ2: $1 \mid \text{CONDW}, \sum_{j=1}^n u_j \leq U \mid Z_2$

The classical assignment and scheduling problem with common due-window on single machine was first introduced by [15], and since then studied in numerous papers as evident by the survey of Janiak et al. [11]. Based on preceding papers, [11] propose that "There is a variety of applications of due window scheduling models in JIT manufacturing,

Table 1 Job-position workloads for Numerical examples 3.2.1–3.4.1

Job	Position							
	1	2	3	4	5	6	7	8
1	26	47	19	14	21	23	12	50
2	17	20	33	14	1	50	48	9
3	39	17	50	41	37	11	12	8
4	37	32	29	27	21	29	38	22
5	20	15	24	12	20	3	4	7
6	10	28	31	25	37	12	11	5
7	48	49	6	22	18	2	43	21
8	36	44	25	28	2	17	30	20

Table 2 Solution for Numerical example 3.2.1—optimal sequence, resource allocation, actual processing times, and cost

ξ_r	0.0	3.0	6.0	9.0	12.0	12.0	8.0	4.0
φ^*	4	3	8	1	2	7	5	6
w_{jr}	37.000	17.000	25.000	14.000	1.000	2.000	4.000	5.000
u_j^*	0.000	1.116	2.014	2.175	1.093	1.378	1.325	0.899
p_{jr}^*	37.000	3.903	3.523	2.537	0.956	1.205	1.738	2.358
Z^*	104.950							

semi-conductor manufacturing, chemical processing, PERT/CPM scheduling as well as in IT". More recent articles include [14, 16, 17, 19, 27–29, 32, 36, 37, 40, 42].

The following properties are known in scheduling theory for the classic problem presented by [15], i.e., with constant processing times, and are still valid for our problem.

Property 3.3.1 An optimal schedule exists in which $d_1 = C_l$ for some l , and $d_2 = C_m$ for some m , such that $m \geq l$.

Property 3.3.2 An optimal schedule exists in which the indices of the jobs defined in Property 3.3.1 are:

$$l = \lceil \frac{n(\delta - \gamma)}{\alpha} \rceil, \quad (19)$$

$$m = \lceil \frac{n(\beta - \delta)}{\beta} \rceil. \quad (20)$$

Property 3.3.3 The positional weights are given by:

$$\xi_r = \begin{cases} \alpha(r-1) + n\gamma, & r = 1, \dots, l \\ n\delta, & r = l+1, \dots, m \\ \beta(n-r+1), & r = m+1, \dots, n \end{cases}. \quad (21)$$

From Property 3.3.1, the optimal due-dates are given by:

$$d_1 = \sum_{j=1}^l p_{jr}. \quad (22)$$

Table 3 Solution for Numerical example 3.3.1—optimal sequence, resource allocation, actual processing times, common due-dates, and cost

ξ_r	24.0	35.0	46.0	56.0	56.0	56.0	40.0	20.0
φ^*	6	3	8	1	2	7	5	4
w_{jr}	10.0	17.0	25.0	14.0	1.0	2.0	4.0	22.0
u_j^*	0.941	1.444	1.970	1.851	0.768	0.968	0.974	1.083
p_{jr}^*	3.260	3.431	3.562	2.750	1.141	1.438	2.026	4.506
d_1^*	10.254							
d_2^*	15.582							
Z^*	831.763							

$$d_2 = \sum_{j=1}^m p_{jr}. \quad (23)$$

Given Properties 3.3.1–3.3.3, applying the unified approach is straightforward.

Algorithm 3.3.1

1. Use Eqs. (19)–(21) to determine l, m and $\xi_r, r = 1, \dots, n$, respectively.
2. Apply LAP 3.1.1 to obtain an optimal sequence.
3. Utilize Eq. (8) to compute W .
4. Use Eq. (9) to obtain $u_j, j = 1, \dots, n$.
5. Apply Eq. (10) to determine $p_{jr}, j = 1, \dots, n$.
6. Utilize Eqs. (22) and (23) to calculate the optimal common due-dates, d_1^* and d_2^* .
7. Use Eq. (12) to find Z_2^* .

Numerical example 3.3.1 We solve the 8×8 job-position workloads problem given in Table 1, with the total amount of resource $U = 10$ and $k = 0.5$. The cost parameters are $\alpha = 11, \beta = 20, \gamma = 3$ and $\delta = 7$.

We use Algorithm 3.3.1 to solve the problem. From (19) and (20), the relevant indices are: $l = 3$ and $m = 6$, respectively. Jobs 6, 3 and 8 are early, jobs 1, 2 and 7 are completed within the due-window, and jobs 5 and 4 are late. The summary of the solution is given in Table 3.

3.4 Problem PZ3: $\left| SLKDW, \sum_{j=1}^n u_j \leq U \right| Z_3$

The due-date assignment and scheduling problem based on flow-allowance (slack, SLK) method was introduced by [1] and later was generalized by [23] to slack with due-window setting (SLKDW). In this subsection, we demonstrate the application of the unified scheme to solve the SLKDW problem. Unlike the CONDW method, in SLKDW the job-dependent due-dates can be expressed as a simple linear function of the job processing times. Generally, each of the due-dates is defined as the sum of its processing time and a job-independent constant. Significant papers considering different SLK objective functions include: [4, 12, 14, 22, 30, 31, 33, 35, 37, 40].

As with the discussion in Sect. 3.3 for the CONDW method, we start with several important and applicable results presented for the constant processing time problem. [23] and [33] proved the following properties for the due-window based on common flow-allowance problem.

Property 3.4.1 *An optimal schedule exists in which $q_1 = C_l$ for some l , and $q_2 = C_m$ for some m , such that $m \geq l$.*

Property 3.4.2 *An optimal schedule exists in which the indices of the jobs defined in Property 3.4.1 are:*

$$l = \lceil \frac{n(\delta - \gamma)}{\alpha} \rceil, \quad (24)$$

$$m = \lceil \frac{n(\beta - \delta)}{\beta} \rceil. \quad (25)$$

Property 3.4.3 *The positional weights are given by:*

$$\xi_r = \begin{cases} \alpha r + n\gamma, & r = 1, \dots, l \\ n\delta, & r = l + 1, \dots, m \\ \beta(n - r), & r = m + 1, \dots, n \end{cases}. \quad (26)$$

Since the positional weight of the job assigned to the last position is zero, we do not allocate resource to this job, and present the following properties:

Property 3.4.4 *The total available resource should be allocated to the jobs in positions $1, \dots, n - 1$.*

Analogous to problem PZ1, once we find the optimal sequence the following is adequate:

$$\text{i. } u_j = 0 \text{ if } X_{jn} = 1. \quad (27)$$

$$\text{ii. } p_{jn} = w_{jn} \text{ if } X_{jn} = 1. \quad (28)$$

iii. The Eqs. (8) and (9), given in Sect. 3.2, must be modified as follows,

$$W = \sum_{j=1}^n \sum_{r=1}^{n-1} \xi_r^{\frac{1}{1+k}} w_{jr}^{\frac{k}{1+k}} X_{jr}. \quad (29)$$

$$u_j = \frac{U}{W} \xi_r^{\frac{1}{1+k}} w_{jr}^{\frac{k}{1+k}}, j = 1, \dots, n, \quad r = 1, \dots, n - 1. \quad (30)$$

We use Property 3.4.1 to find the equations for the optimal flow-allowance constants:

$$q_1 = \sum_{j=1}^l p_{jr}. \quad (31)$$

$$q_2 = \sum_{j=1}^m p_{jr}. \quad (32)$$

Algorithm 3.4.1

Step 1: Utilize Eqs. (24)–(26) to calculate l, m and $\xi_r, r = 1, \dots, n$, respectively.

Step 2: Apply LAP 3.1.1 to obtain an optimal sequence.

Step 3: If $X_{jn} = 1$, then $u_{jn} = 0$ and $p_{jn} = w_{jn}$.

Step 4: Use (29) to calculate W .

Step 5: Utilize (30) to obtain $u_j, j = 1, \dots, n, r = 1, \dots, n - 1$.

Step 6: Apply (10) to compute $p_{jr}, j = 1, \dots, n, r = 1, \dots, n - 1$.

Step 7: Use (31) and (32) to calculate the optimal common flow-allowance constants, q_1^* and q_2^* .

Step 8: Use (12) to calculate Z_3^* .

Numerical example 3.4.1 We solve a problem with the same input data as assumed in Numerical example 3.3.1.

The indices of the jobs completed at $q_1 = C_l$ and $q_2 = C_m$ are: $l = 2$ and $m = 5$, implying that jobs 6 and 3 are early, jobs 7, 2 and 8 are completed within the time interval, and jobs 5, 1 and 4 are tardy. The solution is shown in Table 4.

We conclude Sect. 3 by emphasizing that since the due-window problems that we study in Sects. 3.3 and 3.4 are generalizations of the due–due method, clearly problems involving *common due-date* can be solved with very minor modifications to Algorithms 3.3.1 and 3.4.1, respectively.

Table 4 Solution for Numerical example 3.4.1—optimal sequence, resource allocation, actual processing times, common flow-allowance constants, due-dates, and cost

ξ_r	35.0	46.0	56.0	56.0	56.0	40.0	20.0	0.0
φ^*	6	3	7	2	8	5	1	4
w_{jr}	10.0	17.0	6.0	14.0	2.0	3.0	12.0	22.0
u_j^*	1.355	1.940	1.563	2.074	1.084	0.992	0.992	0.0
p_{jr}^*	2.717	2.960	1.959	2.598	1.358	1.739	3.479	22.0
q_1^*	5.676							
q_2^*	11.592							
d_{1j}	8.393	8.636	7.635	8.275	7.035	7.416	9.155	27.676
d_{2j}	14.309	14.552	13.551	14.190	12.950	13.331	15.071	33.592
Z^*	701.654							

1. The common *due-date* problem can be solved by setting the positional weights (see Property 3.3.3) to:

$$\xi_r = \begin{cases} \alpha(r-1) + n\gamma, r = 1, \dots, l \\ \beta(n-r+1), r = l+1, \dots, n \end{cases}$$

The optimal due-date is given by $d = \sum_{j=1}^l p_{jr}$.

2. The common due-date based on *flow-allowance* problem can be solved by modifying the positional weights (Property 3.4.3) to: $\xi_r = \begin{cases} \alpha r + n\gamma, r = 1, \dots, l \\ \beta(n-r), r = l+1, \dots, n \end{cases}$

The optimal flow-allowance can be found by $q = \sum_{j=1}^l p_{jr}$.

Since the main difference is in the equations of the positional weights, the computational complexity is dominated by the running time of LAP 3.1.1, and thus remains $O(n^3)$.

Supplementary to the above solved problems, in the “Appendix” we briefly indicate an additional two groups of scheduling problems that can be solved to optimality by using the unified method.

4 A unified approach for minimizing resource consumption cost

Similar to Sect. 3 in this section, we start with devising the principles of the unified approach to solving scheduling problems involving positional weights. Unlike Sect. 3, we now seek to minimize the total amount of consumed resource given an upper bound on the scheduling measure. In Sect. 4.1, we solve the general problem and in later sub-sections address practical problems.

4.1 Problem PU0:1| $Z_0 \leq Q|U$

Let $U = \sum_{j=1}^n u_j$, and problem PU0 can be formulated as the following NLP, denoted by NLP 4.1.1

NLP 4.1.1

min U

s.t.

$$\sum_{j=1}^n \sum_{r=1}^n \xi_r p_{jr} X_{jr} \leq Q$$

$$u_j \geq 0, j = 1, \dots, n.$$

Similar to NLP 3.1.1, NLP 4.1.1 can be solved to optimality by applying the KKT conditions.

Let φ denote a given sequence and η denote the single Lagrange multiplier, and

$$\phi(\varphi, \mathbf{u}) = \sum_{j=1}^n \sum_{r=1}^n \xi_r p_{jr} X_{jr}.$$

The Lagrangian $\mathcal{L}_2$ for problem PU0 is defined as,

$$\mathcal{L}_2(\varphi, \mathbf{u}, \eta) = U_0(\varphi, \mathbf{u}) + \eta(\phi(\varphi, \mathbf{u}) - Q).$$

The KKT conditions for PU0 are:

$$1. \quad \frac{\partial U(\varphi, \mathbf{u})}{\partial u_j} + \eta \frac{\partial \phi(\varphi, \mathbf{u})}{\partial u_j} \geq 0$$

$$2. \quad u_j^* \left(\frac{\partial U}{\partial u_j}(\varphi, \mathbf{u}) + \eta \frac{\partial \phi(\varphi, \mathbf{u})}{\partial u_j} \right) = 0$$

$$3. \quad \phi(\varphi, \mathbf{u}) - Q \leq 0$$

$$4. \quad \eta(\phi(\varphi, \mathbf{u}) - Q) = 0$$

$$5. \quad u_j \geq 0, j = 1, \dots, n$$

$$6. \quad \eta \geq 0$$

Similar to Sect. 3, we start with a given sequence and seek the optimality conditions.

Assume that for a given sequence φ , the binary variable $X_{hl} = 1$, and we differentiate the Lagrangian $\mathcal{L}_2$ with respect to u_h .

$$\frac{\mathcal{L}_2(\varphi, \mathbf{u}, \eta)}{\partial u_h} = \frac{\partial U(\varphi, \mathbf{u})}{\partial u_h} + \eta \frac{\partial \theta(\varphi, \mathbf{u})}{\partial u_h} = 1 - \eta k \xi_l \frac{1}{u_h} \left(\frac{w_{hl}}{u_h} \right)^k. \quad (33)$$

To satisfy Condition 2, we assume that $u_j > 0, j = 1, \dots, n$, thus equating (33) to zero we obtain an equation for the Lagrange multiplier,

$$\eta = \frac{u_h}{k \xi_l} \left(\frac{u_h}{w_{hl}} \right)^k > 0. \quad (34)$$

Equation (34) is valid for each job j if assigned to position r in φ , i.e., if $X_{jr} = 1$, thus,

$$\eta = \frac{u_j}{k \xi_r} \left(\frac{u_j}{w_{jr}} \right)^k, j = 1, \dots, n, \quad r = 1, \dots, n. \quad (35)$$

Equating (34) and (35) we get:

$$u_j = \left(\frac{\xi_r}{\xi_l} \right)^{\frac{1}{1+k}} \left(\frac{w_{jr}}{w_{hl}} \right)^{\frac{k}{1+k}} u_h, j = 1, \dots, n, \quad r = 1, \dots, n. \quad (36)$$

Next, we seek an equation for the actual processing time of job $j, j = 1, \dots, n$.

$$p_{jr} = \left(\frac{w_{jr}}{u_j} \right)^k, j = 1, \dots, n, \quad r = 1, \dots, n.$$

From Condition 4 and (36),

$$\begin{aligned} Q &= \sum_{j=1}^n \sum_{r=1}^n \xi_r p_{jr} X_{jr} = \sum_{j=1}^n \sum_{r=1}^n \xi_r \left(\frac{w_{jr}}{u_j} \right)^k X_{jr} \\ &= \xi_l^{\frac{k}{1+k}} \left(\frac{1}{u_h} \right)^k w_{hl}^{\frac{k^2}{1+k}} \sum_{j=1}^n \sum_{r=1}^n \xi_r^{\frac{1}{1+k}} w_{jr}^{\frac{k}{1+k}} X_{jr}. \end{aligned} \quad (37)$$

After rearranging terms we obtain the following for the amount of resource allocated to job h ,

$$u_h^k = \frac{1}{Q} \xi_l^{\frac{k}{1+k}} w_{hl}^{\frac{k^2}{1+k}} \sum_{j=1}^n \sum_{r=1}^n \xi_r^{\frac{1}{1+k}} w_{jr}^{\frac{k}{1+k}} X_{jr}. \quad (38)$$

Let,

$$V = \sum_{j=1}^n \sum_{r=1}^n \xi_r^{\frac{1}{1+k}} w_{jr}^{\frac{k}{1+k}} X_{jr}. \quad (39)$$

Thus, the equation for the amount of resource allocated to job $j, j = 1, \dots, n$ is,

$$u_j = \left(\frac{V}{Q} \right)^{\frac{1}{k}} \xi_r^{\frac{1}{1+k}} w_{jr}^{\frac{k}{1+k}} > 0, \quad j, r = 1, \dots, n. \quad (40)$$

Using (40), the actual processing time of job $j, j = 1, \dots, n$ is given by:

$$p_{jr} = \left(\frac{w_{jr}}{u_j} \right)^k = \frac{Q}{V} \xi_r^{-\frac{k}{1+k}} w_{jr}^{\frac{k}{1+k}}, \quad j, r = 1, \dots, n. \quad (41)$$

Now that we have ascertained that all KKT conditions are satisfied, we are ready to introduce an equation for the optimal total amount of consumed resource,

$$U^* = \left(\frac{1}{Q} \right)^{\frac{1}{k}} \sum_{j=1}^n \sum_{r=1}^n V^{\frac{1}{k}} \xi_r^{\frac{1}{1+k}} w_{jr}^{\frac{k}{1+k}} X_{jr}. \quad (42)$$

After the optimal sequence is realized, Eq. (42) can be rewritten in the following concise form,

$$U^* = \left(\frac{V}{Q} \right)^{\frac{1}{k}} V = \left(\frac{1}{Q} \right)^{\frac{1}{k}} V^{\frac{1+k}{k}}. \quad (43)$$

Analyzing (42), we claim that, similar to Sect. 3.1, the optimal sequence that minimizes the total amount of consumed resource can be achieved by minimizing the expression $\sum_{j=1}^n \sum_{r=1}^n \xi_r^{\frac{1}{1+k}} w_{jr}^{\frac{k}{1+k}} X_{jr}$.

Consequently, we can apply the Linear Assignment Problem LAP 3.1.1, provided in Sect. 3, to obtain the optimal sequence.

Algorithm 4.1.1

1. Find the positional weights.
2. Apply LAP 3.1.1 to obtain an optimal sequence.
3. Utilize Eq. (39) to determine V .
4. Apply Eq. (40) to obtain $u_j, j = 1, \dots, n$.
5. Utilize Eq. (41) to calculate $p_{jr}, j, r = 1, \dots, n$.
6. Use Eq. (43) to find the optimal total consumed resource U^* .

Theorem 4.1 The computational complexity of Algorithm 4.1 is $O(n^3)$.

In the following sub-sections, we address the same problems studied in Sects. 3.3–3.4. For the sake of brevity, only the adapted equations and solution algorithms are provided, complemented by numerical examples. Again, as the main difference is manifested in the calculation of the positional weights, our closing discussion

of Sect. 3 is still valid and the computational complexity of the solution procedures remains $O(n^3)$.

4.2 Problem PU1:1|URDD, $Z_1 \leq Q|U$

In this subsection, we concisely summarize the needed modifications to the unified method, to solve problem **PU1** and provide a numerical example.

$$V = \sum_{j=1}^n \sum_{r=2}^n \xi_r^{\frac{1}{1+k}} w_{jr}^{\frac{k}{1+k}} X_{jr}. \quad (44)$$

$$u_j = \left(\frac{V}{Q} \right)^{\frac{1}{k}} \xi_r^{\frac{1}{1+k}} w_{jr}^{\frac{k}{1+k}}, \quad j = 1, \dots, n, \quad r = 2, \dots, n. \quad (45)$$

$$p_{jr} = \left(\frac{w_{jr}}{u_j} \right)^k = \frac{Q}{V} \xi_r^{-\frac{k}{1+k}} w_{jr}^{\frac{k}{1+k}}, \quad j = 1, \dots, n, \quad r = 2, \dots, n. \quad (46)$$

Algorithm 4.2.1

1. Use (13) and (14) to calculate the index of the on-time job and the positional weights, respectively.
2. Apply LAP 3.1.1 to obtain an optimal sequence.
3. If $X_{j1} = 1$, set $u_j = 0$ and $p_{j1} = w_{j1}$.
4. Utilize Eq. (44) to compute V .
5. Utilize Eq. (45) to calculate $u_j, j = 1, \dots, n, r = 2, \dots, n$.
6. Apply Eq. (46) to find $p_{jr}, j = 1, \dots, n, r = 2, \dots, n$.
7. Use (43) to obtain U^* .

Numerical example 4.2.1 We solve the 8×8 job-position workloads problem given in Table 5, such that $d = 100$, $\alpha = 15$ and $\beta = 8$, the upper bound on the scheduling measure is $Q = 750$ and $k = 0.5$.

The optimal solution is shown in Table 6.

4.3 Problem PU2:1|CONDW, $Z_2 \leq Q|U$

Algorithm 4.1.1 with very minor changes can be applied to solve problem **PU2**:

Algorithm 4.3.1

1. Use (19)–(21) to determine l, m and $\xi_r, r = 1, \dots, n$, respectively.
2. Apply LAP 3.1.1 to obtain an optimal sequence.
3. Use (39) to find V .
4. Utilize Eq. (40) to compute $u_j, j = 1, \dots, n$.
5. Use (41) to calculate $p_{jr}, j = 1, \dots, n$.
6. Apply Eqs. (22) and (23) to calculate d_1^* and d_2^* .
7. Use (43) to calculate U^* .

Numerical example 4.3 We solve the 8×8 job-position workloads problem given in Table 2, $Q = 1250$, $k = 0.5$ and the cost parameters are: $\alpha = 10$, $\beta = 17$, $\gamma = 7$ and $\delta = 9$.

The optimal solution is given in Table 7.

Table 5 Job-position workloads for Numerical examples 4.2.1–4.4.1

Job	Position							
	1	2	3	4	5	6	7	8
1	8	22	32	49	34	20	17	39
2	42	15	47	47	39	4	9	14
3	21	21	27	48	18	12	26	35
4	11	7	38	49	9	12	11	23
5	33	30	14	4	9	25	4	50
6	37	48	9	6	34	28	26	24
7	40	37	42	33	26	23	42	44
8	29	49	21	15	17	18	25	2

Table 6 Solution for Numerical example 4.2.1—optimal sequence, resource allocation, actual processing times, and total consumed resource

ξ_r	0.0	15.0	30.0	40.0	32.0	24.0	16.0	8.0
φ^*	7	3	6	5	4	2	1	8
w_{jr}	40.0	21.0	9.0	4.0	9.0	4.0	17.0	2.0
u_j^*	0.000	0.367	0.440	0.406	0.459	0.289	0.357	0.110
p_{jr}^*	40.000	7.561	4.524	3.137	4.428	3.719	6.897	4.258
U^*	2.429							

Table 7 Solution for Numerical example 4.3.1—optimal sequence, resource allocation, actual processing times, common due-dates, and total consumed resource

ξ_r	56.0	66.0	72.0	72.0	68.0	51.0	34.0	17.0
φ^*	1	4	6	5	3	2	7	8
w_{jr}	8.0	7.0	9.0	4.0	18.0	4.0	42.0	2.0
u_j^*	1.028	1.098	1.265	0.965	1.534	0.767	1.282	0.293
p_{jr}^*	2.789	2.525	2.668	2.036	3.426	2.284	5.725	2.614
d_1^*	5.314							
d_2^*	10.018							
U^*	8.231							

4.4 Problem PU3: $1|SLKDW, Z_3 \leq Q|U$

To solve problem **PU3**, the following adaptations of the unified method are necessary:

$$V = \sum_{j=1}^n \sum_{r=1}^{n-1} \xi_r^{\frac{1}{1+k}} w_{jr}^{\frac{k}{1+k}} X_{jr}. \quad (47)$$

$$u_j = \left(\frac{V}{Q} \right)^{\frac{1}{k}} \xi_r^{\frac{1}{1+k}} w_{jr}^{\frac{k}{1+k}}, \quad j = 1, \dots, n, \quad r = 1, \dots, n-1. \quad (48)$$

$$p_{jr} = \left(\frac{w_{jr}}{u_j} \right)^k = \frac{Q}{V} \xi_r^{-\frac{k}{1+k}} w_{jr}^{\frac{k}{1+k}}, \quad j = 1, \dots, n, \quad r = 1, \dots, n-1. \quad (49)$$

Algorithm 4.4.1

1. Use Eqs. (26), (27) and (23) to determine l, m and the positional weights, respectively.
2. Apply LAP 3.1.1 to obtain an optimal sequence.
3. If $X_{jn} = 1$, set $u_j = 0$ and $p_{jn} = w_{jn}$.
4. Utilize Eq. (47) to compute V .
5. Apply Eq. (48) to calculate $u_j, j = 1, \dots, n, r = 1, \dots, n-1$.
6. Utilize Eq. (49) to find $p_{jr}, j = 1, \dots, n, r = 1, \dots, n-1$.
7. Use (31) and (32) to obtain q_1^* and q_2^* .
8. Use (43) to calculate U^* .

Table 8 Solution for Numerical example 4.4.1—optimal sequence, resource allocation, actual processing times, common flow-allowance constants, due-dates, and total consumed resource

ξ_r	66.0	72.0	72.0	68.0	51.0	34.0	17.0	0.0
φ^*	1	4	6	5	8	2	3	7
w_{jr}	8.0	7.0	9.0	4.0	17.0	4.0	26.0	44.0
u_j^*	0.835	0.846	0.920	0.676	0.904	0.426	0.501	0.0
p_{jr}^*	3.096	2.876	3.128	2.433	4.337	3.065	7.207	44.0
q_1^*	3.096							
q_2^*	9.100							
d_{1j}	6.192	5.972	6.224	5.529	7.433	6.161	10.303	47.096
d_{2j}	12.196	11.976	12.228	11.533	13.437	12.165	16.307	53.100
U^*	5.107							

Numerical example 4.4.1 We solve a problem with the same input data designed for numerical example 4.3.1, for which the optimal solution is provided in Table 8.

5 Conclusion

We studied scheduling problems where the objective function is a scalar product of the processing times vector and the positional penalties vector. We considered controllable processing times jobs with convex resource allocation. Specifically, we focused on position-dependent workloads and provided unified methods for two types of problems: minimizing a scheduling measure subject to a constraint on the resource consumption and the complementary problem of minimizing the consumed resource given an upper bound of the scheduling measure. We proved that by applying our solution algorithms each of these cases can be solved to optimality in $O(n^3)$ time. We demonstrated the use of the unified schemes to solve several scheduling measures with earliness and tardiness penalties.

To the best of our knowledge, no study has been carried out on convex resource allocation on proportionate flow-shops, thus a challenging future research might be minimizing the makespan or/and flowtime even with *position-independent* workloads in this setting. Several scheduling problems with *position-dependent processing times* have been studied on identical parallel machines.

Thus, another interesting direction for future research is the extension of these problems to position-dependent workloads.

Acknowledgements This research did not receive any specific Grant from funding agencies in the public, commercial, or not-for-profit sectors.

Compliance with ethical standards

Conflict of interest The authors declare that they have no conflict of interest.

Appendix

In the “Appendix”, we supplement our study by specifying two additional different groups of scheduling problems that can be solved by using the unified method when position-dependent workloads are considered.

Bicriteria of makespan and total completion time

Oron [24] studied minimizing several scheduling measures on a single machine with position-dependent workloads and learning effect. The addressed objective functions were the makespan (C_{max}), i.e., the maximal completion time, the total completion time (TCT) and the linear combination of makespan and total completion, i.e., $\varepsilon C_{max} + (1 - \varepsilon)TCT$, where ε is the weight of the makespan. Using Algorithm 3.1.1, these problems can be solved with general position-dependent workloads, by adapting the following positional weights:

- i. Minimizing the makespan: $\xi_r = 1, r = 1, \dots, n$.
- ii. Minimizing the total completion time: $\xi_r = (n - r + 1), r = 1, \dots, n$.
- iii. Minimizing the linear combination of makespan and total completion: $\xi_r = \varepsilon + (1 - \varepsilon)(n - r + 1), r = 1, \dots, n$.

Bicriteria of TADC and total completion time

Next, we indicate several TADC and TADW problems on a single machine with position-dependent workloads that can be solved quite straightforwardly by using the unified method. The following positional weights were summarized in [21] and should be used when applying Algorithm 3.1.1:

1. Minimizing the TADC: $\xi_r = (r - 1)(n - r + 1), r = 1, \dots, n$.

2. Minimizing the linear combination of TCT and TADC: $\xi_r = \varepsilon(n - r + 1) + (1 - \varepsilon)(r - 1)(n - r + 1), r = 1, \dots, n$.
3. Minimizing the TADW: $\xi_r = r(n - r), r = 1, \dots, n$.
4. Minimizing the linear combination of total waiting time and TADW: $\xi_r = \varepsilon(n - r) + (1 - \varepsilon)r(n - r), r = 1, \dots, n$.

We note that when solving TADC (TADW) problems, extra caution must be exercised since the positional weight of the first (last) job equals zero, implying that no resource should be allocated to this job (see Algorithms 3.2.1 and 3.4.1, respectively).

References

1. Adamopoulos GI, Pappis CP (1996) Single machine scheduling with flow allowances. *J Oper Res Soc* 47:1280–1285
2. Agnetis A, Mosheiov G (2017) Scheduling with job-rejection and position-dependent processing times on proportionate flowshops. *Optim Lett* 11:885–892
3. Baker KR, Scudder GD (1990) Sequencing with earliness and tardiness penalties: a review. *Oper Res* 38:22–36
4. Cheng B, Cheng L (2018) Single machine slack due-window assignment and scheduling of linear time-dependent deteriorating jobs and a deteriorating maintenance activity. *Open Access Libr J* 5:1–9
5. Fu Y, Zhou M, Guo X, Qi L (2019) Artificial-molecule-based chemical reaction optimization for flow shop scheduling problem with deteriorating and learning effects. *IEEE Access* 7:53429–53440
6. Gao F, Liu M, Wang J-J, Lu Y-Y (2018) No-wait two-machine permutation flow shop scheduling problem with learning effect, common due date and controllable job processing times. *Int J Prod Res* 56:2361–2369
7. Geng X-N, Wang J-B, Bai D (2019) Common due date assignment scheduling for a no-wait flowshop with convex resource allocation and learning effect. *Eng Optim* 51:1301–1323. <https://doi.org/10.1080/0305215X.2018.1521397>
8. Graham RL, Lawler EL, Lenstra JK (1979) Optimization and approximation in deterministic sequencing and scheduling: a survey. *Ann Discret Math* 4:287–326
9. Hardy GH, Littlewood JE, Polya G (1967) *Inequalities*. Cambridge University Press, London
10. Hillier FS, Lieberman JL (2010) *Introduction to operations research*. McGraw-Hill, New York
11. Janiak A, Janiak WA, Krysiak T, Kwiatkowski T (2015) A survey on scheduling problems with due windows. *Eur J Oper Res* 242:347–357
12. Ji P, Li G, Huo Y, Wang JB (2014) Single-machine common flow allowance scheduling with job-dependent aging effects and a deteriorating maintenance activity. *Optim Lett* 8:1389–1400
13. Leyvand Y, Shabtay D, Steiner G (2010) A unified approach for scheduling with convex resource consumption functions using positional penalties. *Eur J Oper Res* 206:301–312
14. Li G, Luo ML, Zhang W-J, Wang X-Y (2015) Single-machine due-window assignment scheduling based on common flow allowance, learning effect and resource allocation. *Int J Prod Res* 53:1228–1241
15. Liman SD, Panwalkar SS, Thongmee S (1998) Common due window size and location determination in a single machine scheduling problem. *J Oper Res Soc* 49:1007–1010

16. Liu L, Wang JJ, Liu F, Liu M (2017) Single machine due window assignment and resource allocation scheduling problems with learning and general positional effects. *J Manuf Syst* 43:1–14
17. Liu L, Wang JJ, Wang XY (2016) Single machine due-window assignment scheduling with resource-dependent processing times to minimise total resource consumption cost. *Int J Prod Res* 54:1186–1195
18. Lu YY, Liu JY (2018) A note on resource allocation scheduling with position-dependent workloads. *Eng Optim* 50:1810–1827
19. Mor B (2019) Single machine minmax common due-window assignment and scheduling problems with convex resource allocation. *Eng Optim* 7:1251–1267
20. Mor B (2019) Minmax scheduling problems with common due-date and completion time penalty. *J Comb Optim* 38:50–71
21. Mor B, Mosheiov G (2011) Total absolute deviation of job completion times on uniform and unrelated machines. *Comput Oper Res* 38:660–665
22. Mor B, Mosheiov G (2012) Minmax scheduling problems with common flow-allowance. *J Oper Res Soc* 63:1284–1293
23. Mosheiov G, Oron D (2010) Job dependent due-window assignment based on a common flow allowance. *Found Comput Decis Sci* 35:185–195
24. Oron D (2016) Scheduling controllable processing time jobs with position-dependent workloads. *Int J Prod Econ* 173:153–160
25. Shabtay D, Steiner G (2007) A survey of scheduling with controllable processing times. *Discret Appl Math* 155:1643–1666
26. Shabtay D, Zofl M (2018) Single machine scheduling with controllable processing times and an unavailability period to minimize the makespan. *Int J Prod Econ* 198:191–200
27. Shi H-B, Wang J-B (2019) Research on common due window assignment flowshop scheduling with learning effect and resource allocation. *Eng Optim*. <https://doi.org/10.1080/0305215X.2019.1604698>
28. Shi L, Chengxin L (2018) Single-machine scheduling with due-window assignment and convex resource allocation. *Adv Appl Math* 7:446–455
29. Sun LH, Cui K, Chen J, Wang J (2016) Due date assignment and convex resource allocation scheduling with variable job processing times. *Int J Prod Res* 54:3551–3560
30. Sun X, Geng XN, Wang JB, Liu F (2018) Convex resource allocation scheduling in the no-wait flowshop with common flow allowance and learning effect. *Int J Prod Res* 57:1873–1891
31. Wang D, Yin Y, Cheng TCE (2017) A bicriterion approach to common flow allowances due window assignment and scheduling with controllable processing times. *Nav Res Logist* 64:41–63
32. Wang J-B, Zhang B, Li L, Bai D, Feng Y-B (2019) Due-window assignment scheduling problems with position-dependent weights on a single machine. *Eng Optim*. <https://doi.org/10.1080/0305215X.2019.1577411>
33. Wu YB, Wan L, Wang X-Y (2015) Study on due-window assignment scheduling based on common flow allowance. *Int J Prod Econ* 165:155–157
34. Yedidsion L, Shabtay D (2017) The resource dependent assignment problem with a convex agent cost function. *Eur J Oper Res* 261:486–502
35. Yin Y, Cheng TCE, Wu C-C, Cheng S-R (2014) Single-machine due window assignment and scheduling with a common flow allowance and controllable job processing time. *J Oper Res Soc* 65:1–13
36. Yin N (2018) Single machine due window assignment resource allocation scheduling with job-dependent learning effect. *J Appl Math Comput* 56:715–725
37. Yin Y, Wang DJ, Wu CC, Cheng TCE (2016) CON/SLK due date assignment and scheduling on a single machine with two agents. *Nav Res Logist* 63:416–429
38. Yin Y, Liu M, Hao J, Zhou M (2012) Single-machine scheduling with job-position-dependent learning and time-dependent deterioration. *IEEE Trans Syst Man Cybern Part A Syst Hum* 42:192–200
39. Zhang X, Liao L, Zhang W, Cheng TCE, Tan Y, Ji M (2018) Single-machine group scheduling with new models of position-dependent processing times. *Comput Ind Eng* 117:1–5
40. Zhang X, Lin W-C, Wu W-H, Wu C-C (2017) Single-machine common/slack due window assignment problems with linear decreasing processing times. *Eng Optim* 49:1388–1400
41. Zhao C, Hsu C-J, Wu W-H, Cheng S-R, Wu C-C (2016) Note on a unified approach to the single-machine scheduling problem with a deterioration effect and convex resource allocation. *J Manuf Syst* 38:134–140
42. Zhu H, Li M, Zhou Z, You Y (2016) Due-window assignment and scheduling with general position-dependent processing times involving a deteriorating and compressible maintenance activity. *Int J Prod Res* 54:3475–3490

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.