Common Due-Window Problem: Polynomial Algorithms for a Given Processing Sequence

Abhishek Awasthi
Department of Computer Science,
University of Applied Sciences,
Görlitz, Germany
Email: abhishek.awasthi@hszg.de

Jörg Lässig
Department of Computer Science,
University of Applied Sciences,
Görlitz, Germany
Email: joerg.laessig@hszg.de

Oliver Kramer
Department of Computing Science
University of Oldenburg,
Oldenburg, Germany
Email: oliver.kramer@uni-oldenburg.de

Thomas Weise
School of Computer Science and Technology,
University of Science and Technology of China,
Hefei, Anhui, China
Email: tweise@ustc.edu.cn

***Abstract*—The paper considers the Common Due-Window (CDW) problem where a single machine processes a certain number of jobs against a common due-window. Each job possesses different processing times but different and asymmetric earliness and tardiness penalties. The objective of the problem is to find the processing sequence of jobs, their completion times and the position of the given due-window to minimize the total penalty incurred due to tardiness and earliness of the jobs. This work presents exact polynomial algorithms for optimizing a given job sequence for a single machine with the run-time complexity of $O(n^2)$, where n is the number of jobs. We also provide an $O(n)$ algorithm for optimizing the CDW with unit processing times. The algorithms take a sequence consisting of all the jobs $(J_i, i = 1, 2, \ldots, n)$ as input and return the optimal completion times, which offers the minimum possible total penalty for the sequence. Furthermore, we implement our polynomial algorithms in conjunction with Simulated Annealing (SA) to obtain the best processing sequence. We compare our results with that of Biskup and Feldmann [2] for different due-window lengths.**
This is a preview version of the paper [1] (see page 10 for the reference). Read the full piece in the proceedings.

I. Introduction

The Common Due-Window (CDW) scheduling problem involves sequencing and scheduling of jobs over machine(s) against a given common due-window. The objective is to find the position of the due-window of a given length and the job sequence to minimize the total tardiness and earliness penalties. Each job possesses a processing time and different penalties per unit time in case the job is completed before or later than the due-window. The jobs which are completed between or at the due-window are called straddle jobs and do not incur any penalty. Similar to the Common Due-Date (CDD) problem, the CDW also occurs in the supply chain management industry to reduce the earliness and tardiness of the goods produced.

Common due-date problems have been studied extensively during the last 30 years with several variants and special cases [3], [4], [5], [6], [7], [8]. CDW is an extension of the CDD with the presence of a common due-window instead of a common due-date. However, several important similar properties hold for both the problems. In 1994, Krämer and Lee studied the due-window scheduling for the parallel machine case and presented three properties for the CDW [9].

Property 1. *There exists an optimal schedule without machine idle time between the first and the last job.*

Proof: Refer to [9], [10]. ■

Property 2. *In any optimal schedule, jobs completed before d_l (the left due-date) are sequenced in the reverse SPT order, and the jobs which start after d_r (the right due-date) are sequenced in the SPT order.*

Proof: Refer to [9], [10]. ■

Property 3. *Let C_i be the completion time of job i, then there exists an optimal schedule where one of the jobs finishes at d_l or at d_r,* i.e.,

a) $C_i = d_l$ for some i, or
b) $C_i = d_r$ for some i .

Proof: Refer to [9], [10]. ■

Krämer and Lee also showed that the CDW with unit weight case is also NP-complete and provided a dynamic programming algorithm for the two machine case [9]. Liman *et al.* considered the CDW with constant earliness/tardiness penalties and proposed an $O(n \log n)$ algorithm to minimize the weighted sum of earliness, tardiness and due-window location [11]. The same authors also studied the CDW on a single machine with controllable processing times with constant penalties for earliness, tardiness and window location, and different penalties for compression of job processing times. They showed that the problem can be formulated as an assignment problem and can be solved using the well-known algorithms [12].

In 2002, Chen and Lee studied the CDW on parallel machines and solved the problem using a Branch and Bound algorithm and showed that the problem can be solved up to 40 jobs on any number of machines [13] in a reasonable time. In 2005, Biskup and Feldmann dealt with the general case of the CDW problem and approached it with three different metaheuristic algorithms, namely, evolutionary strategy, simulated annealing and threshold accepting. They also validated their approaches on 250 benchmark instances up to 200 jobs [2]. Wan studied the common due-window problem with controllable processing times with constant earliness/tardiness penalties and distinct compression costs, and discussed some properties of the optimal solution along with a polynomial algorithm for the solving the problem in 2007 [14]. Zhao *et al.* studied the CDW with constant earliness/tardiness penalties and window location penalty, and proposed polynomial time approximation schemes [15].

In 2010, Yeung *et al.* formulated a supply chain scheduling control problem involving single supplier and manufacturer and multiple retailers. They formulated the problem as a two machine CDW and presented a pseudo-polynomial algorithm to solve the problem optimally [16]. Cheng *et al.* considered the common due-window assignment problem with time-dependent deteriorating jobs and a deteriorating maintenance activity. They proposed a polynomial algorithm for the problem with linear deterioration penalties and its special cases [17]. Gerstl and Mosheiov studied the due-window assignment problem with unit-time jobs and proposed an $O(n^3)$ algorithm for solving the problem [18].

Yin *et al.* considered the batch delivery single-machine scheduling problem with assignable common due-window with constant penalties and proposed an $O(n^8)$ dynamic programming algorithm under an assumption on the relationship among the cost parameters [19]. In 2013, Janiak *et al.* presented a survey paper on the common due-window assignment scheduling problem and discussed more than 30 different variations of the problem [20]. Again in 2013, Janiak *et al.* studied the CDW assignment problem on parallel machines to minimize the earliness/tardiness penalties along with the penalties associated with the location and size of the due-window [21].

In this paper, we consider the single machine case for the CDW problem with asymmetric penalties for both the general and the unit-time job cases. We make a theoretical study of the CDW problem and present an $O(n^2)$ and $O(n)$ polynomial exact algorithms to optimize a given job sequence on a single machine for the general and unit-time job cases, respectively.

II. Problem Formulation

In this section, we give the mathematical notation of the common due-window problem based on [2]. We also define some new parameters which are later used in the presented algorithms in the next section.

Let
n = the total number of jobs,
d_l = the left common due-date,
d_r = the right common due-date,
d = the common due-window length, $d = d_r - d_l$,
p_i = the processing time of job i, $i = 1, 2, \ldots, n$,
E_i = the earliness of job i,
T_i = the tardiness of job i,
C_i = the completion time of job i,
W_i = the straddle jobs, *i.e.* if $d_l \le C_i \le d_r$, $\forall i$,
α_i = the earliness penalty per unit time for job i,
β_i = the tardiness penalty per unit time for job i.

We can define E_i and T_i mathematically as

$$\left.\begin{aligned} E_i &= \max\{0, d_l - C_i\} \\ T_i &= \max\{0, C_i - d_r\} \end{aligned}\right\} \qquad i = 1, 2, \ldots, n \ .$$

The objective of the problem is to schedule the jobs against the due-window to minimize the total penalty incurred by the earliness and tardiness of the jobs.

$$\min \sum_{i=1}^{n} \{\alpha_i \cdot E_i + \beta_i \cdot T_i\} \ . \tag{1}$$

Before stating the algorithm we first introduce a new vector $DT_i = C_i - d_l$ and $SD_i = C_i - d_r$, $i = 1, 2, \ldots, n$. DT_i and SD_i are just the algebraic deviation of the completion time of any job i from the left and the right due-date, respectively.

Definition 1. *Let PL be a vector of length n where element of PL (PL_i) is the effective penalty possessed by any job i such that*

$$PL_i = \begin{cases} -\alpha_i, & \text{if } DT_i \le 0 \\ \beta_i, & \text{if } SD_i > 0 \\ 0, & \text{otherwise} \ . \end{cases} \tag{2}$$

We also define two new vectors, D and S, to express the objective function mentioned in Equation (1) in a compact form. D_i and S_i are defined for all $i, i = 1, 2, \ldots, n$ such that

$$D_i = \begin{cases} DT_i, & \text{if } DT_i \le 0 \\ 0, & \text{if } DT_i > 0 \ , \end{cases} \tag{3}$$

$$S_i = \begin{cases} 0, & \text{if } SD_i < 0 \\ SD_i, & \text{if } SD_i \ge 0 \ . \end{cases} \tag{4}$$

With the above definitions we can now express the objective function stated by Equation (1) as $\min(Sol)$, where

$$Sol = \sum_{i=1}^{n} \{(D_i + S_i) \cdot PL_i\} \ . \tag{5}$$

We now present and prove an important property for the Common Due-Date problem. Later on, we extend this property for the Common Due Window problem

III. Novel Property for the Common Due-Window Date

Theorem 1. *If the optimal due-date position in any given job sequence of the CDD lies between C_{r-1} and C_r,* i.e., *$C_{r-1} < d \le C_r$, then the following relations hold for the two cases*
Case 1: If $C_{r-1} < d < C_r$

i) $\sum\limits_{i=k+1}^{n} \beta_i \le \sum\limits_{i=1}^{k} \alpha_i, \ k = r, r+1, \ldots, n \ .$

Case 2: If $C_r = d$

i) $\sum_{i=k+1}^{n} \beta_i \leq \sum_{i=1}^{k} \alpha_i,\ k = r, r+1, \ldots, n$ *and*

ii) $\sum_{i=1}^{k} \alpha_i \leq \sum_{i=k+1}^{n} \beta_i,\ k = 1, 2, 3, \ldots, r-1$.

Proof: We know from the property proved by [22] that the optimal schedule of the CDD for any job sequence either has $t^* = 0$ or one of the job finishes at the due-date. Hence, we consider these two cases separately.

Case 1: Optimal schedule with $C_{r-1} < d < C_r$

Let us first consider the case when the optimal schedule for any sequence lies strictly between C_{r-1} and C_r, *i.e.* $C_{r-1} < d < C_r$, as shown in Figure 1. We know from [22] that such a case can occur only when the first job starts at time $t = 0$ and all the following jobs are processed without any machine idle time. Let the difference between C_{r-1} and d in Figure 1 be y such that $y = d - C_{r-1}$. Let E_i and T_i be the earliness and tardiness penalties of any job i, for this particular case, respectively. Hence, the solution value Sol_d for the schedule in Figure 1 can be written as

$$Sol_d = \sum_{i=1}^{r-1} E_i \cdot \alpha_i + \sum_{i=r}^{n} T_i \cdot \beta_i \ . \tag{6}$$

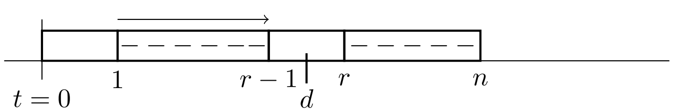

Fig. 1. Assume that the first job starts at time $t = 0$ and the due-date lies between the completion times of two consecutive jobs, with $y = d - C_{r-1}$.

Now, the only possibility to get another schedule is to shift all the jobs to the right such that one of the jobs finishes at the due-date. Figure 2 shows the right shift of all the jobs by y units. It is clear that after this right shift of all the jobs, job $r-1$ offers no penalty. Hence, the earliness of the early jobs in Figure 2 will be $E_i - y$ for $i = 1, 2, \ldots, r-2$ and the tardiness of the tardy jobs will be $T_i + y$ for $i = r, r+1, \ldots, n$. We can now write the solution value for Figure 2 as Sol'_d where

$$Sol'_d = \sum_{i=1}^{r-2} (E_i - y) \cdot \alpha_i + \sum_{i=r}^{n} (T_i + y) \cdot \beta_i \ . \tag{7}$$

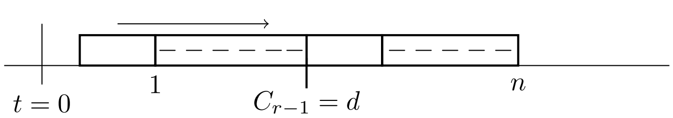

Fig. 2. Assume that the $(r-1)$th job finishes at the due-date d in the optimal schedule.

Since we already assumed that Figure 1 is the optimal schedule, we have

$$Sol_d \leq Sol'_d \ . \tag{8}$$

Note that in Figure 1 the earliness of job r is y. Hence Sol_d can be rewritten as

$$Sol_d = \sum_{i=1}^{r-2} E_i \cdot \alpha_i + y \cdot \alpha_{r-1} + \sum_{i=r}^{n} T_i \cdot \beta_i \ . \tag{9}$$

Likewise, the terms in Sol'_d can also be expanded as

$$Sol'_d = \sum_{i=1}^{r-2} E_i \cdot \alpha_i - \sum_{i=1}^{r-2} y \cdot \alpha_i + \sum_{i=r}^{n} T_i \cdot \beta_i + \sum_{i=r}^{n} y \cdot \beta_i \ . \tag{10}$$

Substituting the value of Sol_d from Equation (9) and Sol'_d from Equation (10) in Equation (8), we get

$$\begin{aligned} y \cdot \alpha_{r-1} &\leq -\sum_{i=1}^{r-2} y \cdot \alpha_i + \sum_{i=r}^{n} y \cdot \beta_i \ , \\ \sum_{i=1}^{r-1} y \cdot \alpha_i &\leq \sum_{i=r}^{n} y \cdot \beta_i \ . \end{aligned} \tag{11}$$

Since $y > 0$ due to case constraint, Equation (11) fetches us

$$\sum_{i=1}^{r-1} \alpha_i \leq \sum_{i=r}^{n} \beta_i \ . \tag{12}$$

Clearly, if Equation 12 holds for any $k = r$, then it will also hold for any $k < r$, since α_i and β_i are positive for all i, $i = 1, 2, \ldots, n$. This proves the first case of Theorem 1.

Case 2: Optimal schedule at $C_r = d$

In this case we assume that the optimal solution lies at the completion time of some job r. Consider Figure 3, where the optimal schedule occurs with the due-date position at the completion time of job r, *i.e.* $C_r = d$.

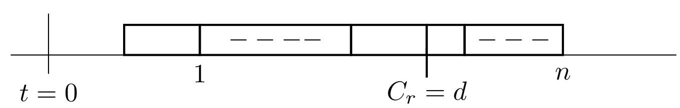

Fig. 3. Assume that the rth job finishes at the due-date d in the optimal schedule.

Let, E_i and T_i be the earliness and tardiness of any job i, respectively, for this particular case (Figure 3) and the solution value for this case be Sol_r, then using Equation (5) we have

$$Sol_r = \sum_{i=1}^{r-1} E_i \cdot \alpha_i + \sum_{i=r+1}^{n} T_i \cdot \beta_i \ . \tag{13}$$

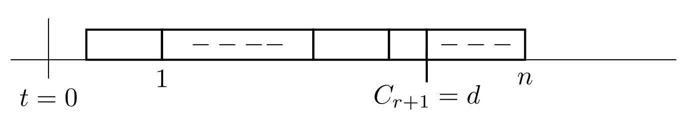

Fig. 4. Schedule with the completion time of job $r+1$ lying at the due-date, $C_{r+1} = d$.

Let the solution value for the case when all the jobs are shifted to the left by p_{r+1}, *i.e.*, the $(r+1)$th job ends at the due-date be Sol_{r+1}, see Figure 4. Then the earliness of jobs 1 to $r-1$ will increase by the processing time of job $r+1$, compared to Figure 3, since the due-date position shifts to right by the same amount and job r will be early by p_{r+1}. Besides, job

$r+1$ offers no penalty and the tardiness of the all the jobs from $r+2$ to n reduces by p_{r+1}. Hence, the objective function value when the due-date is situated at C_{r+1} becomes

$$Sol_{r+1} = \sum_{i=1}^{r-1}(E_i+p_{r+1})\cdot\alpha_i + p_{r+1}\cdot\alpha_r + \sum_{i=r+2}^{n}(T_i-p_{r+1})\cdot\beta_i \ . \tag{14}$$

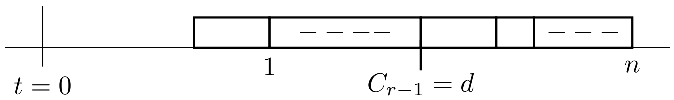

Fig. 5. Schedule with the completion time of job $r-1$ lying at the due-date, $C_{r-1}=d$.

Likewise, when all the jobs are shifted to the right such that the $(r-1)$th job finishes at the due-date, in comparison to Figure 3, then jobs 1 to $r-2$ will have their earliness reduced by p_r, job r will be tardy by p_r and the all the jobs from $r+1$ to n will have their tardiness increased by p_r. Let the solution value for Figure 5 where the $(r-1)$th job ends at the due-date be Sol_{r-1}, then

$$Sol_{r-1} = \sum_{i=1}^{r-2}(E_i-p_r)\cdot\alpha_i + p_r\cdot\beta_r + \sum_{i=r+1}^{n}(T_i+p_r)\cdot\beta_i \ . \tag{15}$$

Since we assume that Sol_r is the optimal value, we have,

$$Sol_r \quad \leq \quad Sol_{r+1} \ , \tag{16}$$

and

$$Sol_r \quad \leq \quad Sol_{r-1} \ . \tag{17}$$

Notice that in the first case, when $C_r = d$, the tardiness of job $r+1$ is p_{r+1} and the earliness of job $r-1$ is p_r. Hence, rearranging the terms in Sol_r we get,

$$Sol_r = \sum_{i=1}^{r-1} E_i\cdot\alpha_i + p_{r+1}\cdot\beta_{r+1} + \sum_{i=r+2}^{n} T_i\cdot\beta_i \ . \tag{18}$$

Splitting the earliness penalty of job $r-1$, Sol_r can also be expressed as,

$$Sol_r = \sum_{i=1}^{r-2} E_i\cdot\alpha_i + p_r\cdot\alpha_{r-1} + \sum_{i=r+1}^{n} T_i\cdot\beta_i \ . \tag{19}$$

Substituting the values of Sol_r from Equation (18) and Sol_{r+1} from Equation (14) in Equation (16) we get,

$$\begin{aligned} Sol_r \quad &\leq \quad Sol_{r+1} \ , \\ \sum_{i=r+1}^{n} p_{r+1}\cdot\beta_i \quad &\leq \quad \sum_{i=1}^{r} p_{r+1}\cdot\alpha_i \text{ and} \\ \sum_{i=r+1}^{n} \beta_i \quad &\leq \quad \sum_{i=1}^{r} \alpha_i \ . \end{aligned} \tag{20}$$

Likewise, substituting the values of Sol_r from Equation (19) and Sol_{r-1} from Equation (15) in Equation (17),

$$\begin{aligned} Sol_r \quad &\leq \quad Sol_{r-1} \ , \\ \sum_{i=1}^{r-1} p_r\cdot\alpha_i \quad &\leq \quad \sum_{i=r}^{n} p_r\cdot\beta_i \text{ and} \\ \sum_{i=1}^{r-1} \alpha_i \quad &\leq \quad \sum_{i=r}^{n} \beta_i \ . \end{aligned} \tag{21}$$

Since α_i and β_i are positive for all i, Equation (20) also implies,

$$\sum_{i=k+1}^{n} \beta_i \leq \sum_{i=1}^{k} \alpha_i, \qquad k = r, r+1, \ldots, n \ , \tag{22}$$

i.e., if the sum of the tardiness penalties for the jobs $(r+1)$ to n is less than the sum of the earliness penalties for the jobs from 1 to r, then the same inequality also holds for any $k \geq r$, since $\beta_i > 0$ and $\alpha_i > 0$ for $i = 1, 2, \ldots, n$. Likewise, Equation (21) implies that

$$\sum_{i=1}^{k} \alpha_i \leq \sum_{i=k+1}^{n} \beta_i, \qquad k = 1, 2, \ldots, r-1 \ , \tag{23}$$

i.e., if the sum of the earliness penalties for the jobs 1 to $(r-1)$ is less than the sum of the tardiness penalties for the jobs from r to n, then the same inequality also holds for any $k \leq r-1$, since $\beta_i > 0$ and $\alpha_i > 0$ for $i = 1, 2, \ldots, n$. This proves the second case of Theorem 1. ■

Since there is only one way that the due-date position may be between the completion times of two consecutive jobs, we first need to calculate the sum of penalties before and after the due-date such that the first job starts at time zero and all the jobs follow without any idle time. Thereafter, we shift all the jobs towards right as long as the sum of the tardiness of jobs finishing after the due-date is less than or equal to the some of the earliness penalties of all the jobs which complete before the due-date.

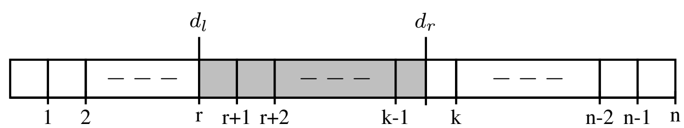

Fig. 6. Schedule for the due-window case, with the left due-date (d_l) situated at C_r and the right due-date (d_r) in between the completion times of jobs k and $(k+1)$.

However, it can be easily be proved that the relationship between the sum of the earliness penalties and the tardiness penalties proved in Theorem 1 for the CDD problem also hold for the CDW problem.

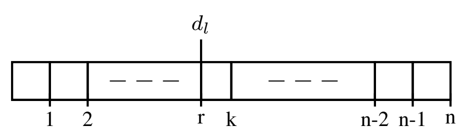

Fig. 7. Schedule for the CDW such that all the straddle jobs removed. The problem now converts to the CDD with the due-date position at C_r.

Theorem 2. *If in the optimal schedule of a CDW instance, jobs $1, 2, \ldots, r-1$ are early and jobs $k, k+1, \ldots, n$, $k > r$ are tardy then, we have, $\sum_{i=k}^{n} \beta_i \leq \sum_{i=1}^{r-1} \alpha_i$ for minimum possible value of k.*

Proof: Using Property 3, let us assume without loss of any generality that in the optimal schedule the left due-date d_l lies at the completion time of a job and the right due-date d_r lies between the completion times of two adjacent jobs.

Figure 6 depicts the optimal position of the due-window for the given instance of n jobs. Jobs $1, 2, \ldots, r-1$ are early and jobs $k, k+1, \ldots, n$ are tardy, then we can discard the straddle jobs (jobs which are within the due-window) and convert the problem to a CDD problem as shown in Figure 7. The problem now converts to the CDD with d_l being the due-date. Hence, the properties of Theorem 1 will also hold for the CDW on the same lines as for the CDD. ■

IV. The Exact Algorithm

Using Theorem 2, we now present our exact polynomial algorithms for the CDW with distinct and unit-time jobs cases. As mentioned above in Properties 1, 2 and 3, we know that the optimal schedule of the CDW has no idle time of the machine between C_1 and C_n. The idea of our algorithm is based on the approaches mentioned in [23], [24]. We first optimize any given sequence using our polynomial algorithm and use a modified Simulated Annealing (SA) algorithm to find the optimal/best processing sequence.

The jobs are initialized with the first job starting at time $t = 0$ and are shifted to the right by $\min\{-DT_{n1}, -SD_{n2}\}$, *i.e.*, minimum deviation of the completion times from the right and the left due-dates. This way, every shift ensure that one of the jobs finishes at one of the due-dates and we do not skip over the optimal position of the due-dates. Once the property mentioned in Theorem 2 is satisfied, we have our optimal schedule and no more shifting is required. We now present Algorithm 1 and 2 to optimize the CDW for the distinct and unit-processing times of the jobs, for a given processing sequence.

V. Proof of Optimality

We now provide the optimality of Algorithm 1 and 2 with respect to the solution value.

Theorem 3. *Algorithm 1 and 2 are optimal for the given sequence with respect to the objective function value.*

Proof: We first schedule a given job sequence such that the processing of the jobs starts at time $t = 0$ and move the jobs towards the right, *i.e.*, increasing the overall tardiness penalty and decreasing the overall earliness penalty, to find the minimum value of k as mentioned in Theorem 2. We increase the completion times by $\min\{DT_i, SD_j\}$ (DT and SD are defined in Section II), where job i possesses the least earliness from d_l and job j possesses the least earliness from d_j. The reason behind performing this operation is due to the Property 2, *i.e.*, either of d_l or d_r can lie at the completion time of a job. After every right shift, the values of DT and SD are updated for the next iteration as long as the property mentioned in Theorem 2 holds. For the case when the jobs are all of unit processing time, then $\min\{DT_i, SD_j\}$ is always equal to one and we do not need to update the two vectors but only sum up the earliness and tardiness penalties to check for Theorem 2. ■

VI. Algorithm Run-Time Complexity

In this section we study and prove the run-time complexity of Algorithms 1 and 2.

Algorithm 1: Exact algorithm for the general CDW for a given job sequence with a run-time complexity of $O(n^2)$.

$C_i \leftarrow \sum_{k=1}^{i} p_k \;\forall\; i = 1, 2, \ldots, n$
Calculate $DT_i, SD_i \;\forall\; i$
$A_i \leftarrow DT_i \;\forall\; i$
$B_i \leftarrow SD_i \;\forall\; i$
$loop \leftarrow 1$
while $loop \leq n$ **do**
    $n1 \leftarrow \underset{i=1,2,\ldots,n}{\arg\max}(DT_i < 0)$
    $n2 \leftarrow \underset{i=1,2,\ldots,n}{\arg\max}(SD_i < 0)$
    $\tau = \min\{-DT_{n1}, -SD_{n2}\}$
    if $(\tau \neq \varnothing)$ **then**
        $DT_i = DT_i + \tau \;\forall\; i$
        $SD_i = SD_i + \tau \;\forall\; i$
        $i_1 \leftarrow \underset{i=1,2,\ldots,n}{\arg\max}(SD_i > 0)$
        $i_2 \leftarrow \underset{i=1,2,\ldots,n}{\arg\max}(DT_i \leq 0)$
        $pe \leftarrow \sum_{i=1}^{i_2} \alpha_i$
        $pl \leftarrow \sum_{i=i_1}^{n} \beta_i$
        if $(pl < pe)$ **then**
            $A_i \leftarrow DT_i \;\forall\; i$
            $B_i \leftarrow SD_i \;\forall\; i$
    $loop \leftarrow loop + 1$
$DT_i \leftarrow A_i \;\forall\; i$
$SD_i \leftarrow B_i \;\forall\; i$
Calculate $D_i, S_i, PL_i \;\forall i$
$Sol \leftarrow \sum_{i=1}^{n}\{(D_i + S_i) \cdot PL_i\}$
return Sol

Algorithm 2: Linear algorithm for a given sequence for the CDW with unit-time job processing times.

$C_i \leftarrow \sum_{k=1}^{i} p_k \;\forall\; i = 1, 2, \ldots, n$
$DT_i \leftarrow C_i - d_l \;\forall\; i$
$SD_i \leftarrow C_i - d_r \;\forall\; i$
$n1 \leftarrow \underset{i=1,2,\ldots,n}{\arg\max}(DT_i < 0)$
$n2 \leftarrow \underset{i=1,2,\ldots,n}{\arg\max}(SD_i < 0)$
$pe \leftarrow \sum_{i=1}^{n1} \alpha_i$
$pl \leftarrow \sum_{i=n2}^{n} \beta_i$
$t \leftarrow 0$
$loop \leftarrow 1$
while $loop \leq n$ **do**
    $n2 \leftarrow n2 - 1$
    $pe \leftarrow pe - \alpha_{n1}$
    $pl \leftarrow pl + \beta_{n2}$
    $n1 \leftarrow n1 - 1$
    if $(pl < pe)$ **then**
        $t \leftarrow loop$
    $loop \leftarrow loop + 1$
$DT_i = DT_i + t \;\forall i$
$SD_i = SD_i + t \;\forall i$
Calculate $D_i, S_i, PL_i \;\forall i$
$Sol \leftarrow \sum_{i=1}^{n}\{(D_i + S_i) \cdot PL_i\}$
return Sol

TABLE I. Results obtained for single machine common due-window problem till 50 jobs. For each job there are 10 different instances each with a value for k and for each k there are 5 different due-windows.

k	h1 - h2	10*		20		50	
		BR	Algo 1+SA	BR	Algo 1+SA	BR	Algo 1+SA
1	0.1 - 0.2	1896	1896	4089	4089	39250	39461
	0.1 - 0.3	1330	1330	2713	2713	28225	28225
	0.2 - 0.5	540	540	1162	1162	12756	12754
	0.3 - 0.4	919	919	2294	2294	21137	21115
	0.3 - 0.5	587	587	1559	1559	14002	13971
2	0.1 - 0.2	947	947	8251	8251	29110	29054
	0.1 - 0.3	539	539	5950	5950	20133	20133
	0.2 - 0.5	191	191	2770	2770	8480	8468
	0.3 - 0.4	432	432	4482	4482	15166	15152
	0.3 - 0.5	265	265	2923	2923	9436	9434
3	0.1 - 0.2	1488	1488	5881	5881	33407	33180
	0.1 - 0.3	1012	1012	4067	4067	23027	23021
	0.2 - 0.5	398	398	1675	1675	9935	9969
	0.3 - 0.4	760	760	3035	3035	17640	17535
	0.3 - 0.5	462	462	1998	1998	11402	11401
4	0.1 - 0.2	2128	2128	8977	8977	25869	25860
	0.1 - 0.3	1576	1576	6609	6609	17568	17544
	0.2 - 0.5	712	712	3113	3113	7378	7376
	0.3 - 0.4	1162	1162	4832	4830	13633	13619
	0.3 - 0.5	740	740	3210	3210	8448	8446
5	0.1 - 0.2	1150	1150	4028	4028	31468	31456
	0.1 - 0.3	755	755	2850	2850	21693	21689
	0.2 - 0.5	284	284	1192	1192	8954	8948
	0.3 - 0.4	542	542	2112	2112	15767	15747
	0.3 - 0.5	339	339	1341	1341	9994	9965
6	0.1 - 0.2	1479	1479	6306	6306	33452	33452
	0.1 - 0.3	1023	1023	4247	4247	23267	23261
	0.2 - 0.5	439	439	1557	1557	10245	10221
	0.3 - 0.4	779	779	3042	3042	17400	17392
	0.3 - 0.5	500	500	1778	1778	11207	11200
7	0.1 - 0.2	2093	2093	10204	10204	42257	42234
	0.1 - 0.3	1521	1521	7492	7492	29277	29274
	0.2 - 0.5	717	717	3573	3573	12014	12000
	0.3 - 0.4	1190	1190	5722	5722	20718	20715
	0.3 - 0.5	809	809	3846	3846	12953	12935
8	0.1 - 0.2	1644	1644	3749	3742	42220	42218
	0.1 - 0.3	1287	1287	2519	2519	28411	28403
	0.2 - 0.5	670	670	991	990	11167	11154
	0.3 - 0.4	952	952	1801	1801	21014	20965
	0.3 - 0.5	680	680	1069	1069	12917	12913
9	0.1 - 0.2	1466	1466	3317	3317	33222	33222
	0.1 - 0.3	1121	1121	2342	2342	23848	23840
	0.2 - 0.5	492	492	1056	1056	10987	10985
	0.3 - 0.4	772	772	1767	1767	17999	17972
	0.3 - 0.5	513	513	1187	1187	11951	11935
10	0.1 - 0.2	1835	1835	4673	4673	31492	31492
	0.1 - 0.3	1384	1384	3266	3266	22056	22040
	0.2 - 0.5	691	691	1355	1355	9653	9653
	0.3 - 0.4	1047	1047	2419	2419	16538	16510
	0.3 - 0.5	717	717	1474	1474	10628	10597

TABLE II. RESULTS OBTAINED FOR SINGLE MACHINE COMMON DUE-WINDOW PROBLEM TILL 50 JOBS, WHERE EVERY JOB HAS A UNIT PROCESSING TIME. FOR EACH JOB THERE ARE 10 DIFFERENT INSTANCES EACH WITH A VALUE FOR k AND FOR EACH k THERE ARE 5 DIFFERENT DUE-WINDOWS.

n	h1 - h2	k=1	k=2	k=3	k=4	k=5	k=6	k=7	k=8	k=9	k=10
	0.1 - 0.2	170	97	157	181	137	204	250	223	201	162
	0.1 - 0.3	115	58	109	126	98	144	179	158	140	118
10	0.2 - 0.5	45	19	46	58	46	66	84	68	62	54
	0.3 - 0.4	77	45	81	92	75	108	145	112	105	86
	0.3 - 0.5	48	27	53	61	49	73	99	72	71	57
	0.1 - 0.2	542	865	647	805	589	858	819	467	622	624
	0.1 - 0.3	372	627	446	565	411	592	587	305	439	426
20	0.2 - 0.5	157	291	185	234	177	235	266	115	203	169
	0.3 - 0.4	286	460	339	397	300	428	441	216	346	310
	0.3 - 0.5	185	306	222	250	197	266	294	130	236	194
	0.1 - 0.2	4641	3935	3769	3981	3341	3747	4327	4124	4100	4053
	0.1 - 0.3	3290	2719	2539	2752	2251	2632	2956	2828	2822	2849
50	0.2 - 0.5	1462	1137	1027	1163	886	1171	1156	1151	1174	1267
	0.3 - 0.4	2460	1978	1834	2075	1601	2006	2050	2077	2052	2180
	0.3 - 0.5	1615	1242	1149	1313	994	1286	1260	1300	1327	1420

Lemma 1. *The run-time complexities of Algorithm 1 and 2 are $O(n^2)$ and $O(n)$, respectively, where n is the total number of jobs.*

Proof: It can easily observed that the complexity of the Algorithm 1 is dependent on the $while$ loop in line 6. The computations of $n1, n2, DT, SD, i_1$ and i_2 are all of $O(n)$ and are computed n times, in the worst case. Hence, the complexity of Algorithm 1 is $O(n^2)$. However, in Algorithm 2 we do not need to compute these parameters after every iteration of the $while$ loop. All the computations inside both the $while$ loops are of $O(n)$ and so are the calculations of $n1, n2, SD, DT, PL$ and Sol. Hence, the run-time complexity of Algorithm 2 is $O(n)$. ■

VII. RESULTS

We now present our computational results for the two cases of CDW, discussed in the paper. We use the CDW benchmark instances provided by Biskup and Feldmann in [2] and compare our results with theirs.

As described in [23], [24], [25], we use a modified Simulated Annealing algorithm to generate job sequences and Algorithm 1 and 2 to optimize each sequence to its minimum penalty for the two cases. Our experiments over all the instances suggest that an ensemble size of $4 + n/10$ and the maximum number of iterations of $500 \cdot n$, where n is the number of jobs, work best for the provided instances in general. The runtime for all the results is the time after which the solutions mentioned in Table I are obtained on average after 10 different replication. The initial temperature is kept as twice the standard deviation of the energy at infinite temperature: $\sigma_{E_{T=\infty}} = \sqrt{\langle E^2 \rangle_{T=\infty} - \langle E \rangle^2_{T=\infty}}$. We estimate this quantity by randomly sampling the configuration space [26]. An exponential schedule for cooling is adopted with a cooling rate of 0.9999 with the Metropolis acceptance criterion, $\min\{1, \exp((-\triangle E)/T)\}$ [26].

The modification from the standard SA is the increase in the temperature after the annealing temperature becomes less than 1 unit. In such a case, we increase the temperature to $1/10$th of the initial temperature. Apart from this, we also incorporate elitism in our SA. Elitism has been successfully adopted in evolutionary algorithms for several complex optimization problems [27], [28]. Theoretical studies have been made analysing speed-ups in parallel evolutionary algorithms combinatorial optimization problems in [29], [30]. We observed that this concept works well for the CDD and the Aircraft Landing Problem problem [23], [24]. As for the perturbation rule, we first randomly select a certain number of jobs in any job sequence and permute them randomly to create a new sequence. The number of jobs selected for this permutation is taken as $3 + \lfloor \sqrt{n/5} \rfloor$, where n is the number of jobs. For large instances the size of this permutation is quite small, but we have observed that it works well with our modified simulated annealing algorithm.

In Table I we present our results for the CDW where the due-window size for any instance is calculated using the values of $h1$ and $h2$. A due-window has a left (d_l) and right (d_r) due-date, where $d_l = \lfloor h1 \cdot \sum_{i=1}^{n} p_i \rfloor$ and $d_r = \lfloor h2 \cdot \sum_{i=1}^{n} p_i \rfloor$, as described in [2]. For the first 50 instances with 10 jobs we obtain the optimal solution for all the instances. For the remaining 100 instances, we achieve better results for 43 instances than Biskup and Feldmann [2], equal results for 55 and for only two instances we do not reach the best known solution value but are within a percentage gap of 0.537 and 0.342.

Table II shows our results for the same instances, but for unit-time processing time of all the jobs. We use Algorithm 2 to optimize any job sequence and the same modified SA to find the best processing sequence. Since, these instances have not been studied for the unit-time processing times, we are unable to compare our results with the literature. Table III shows the run-time in seconds for all the instances for both the cases. The run-times shown are the mean time for all the 10 different instances for each job, averaged over 10 replications of our approach.

VIII. CONCLUSION AND FUTURE DIRECTION

In this paper we present a novel property for the problem of scheduling against a common due-window for the general and the unit processing time cases. We present a theoretical

TABLE III. AVERAGE RUN-TIME IN SECONDS FOR ALL THE 10 DIFFERENT INSTANCES FOR EACH JOB OVER 10 ITERATIONS OF THE ALGORITHMS.

Number of Jobs	Run-time (seconds)	
	Algo 1+SA	**Algo 2+SA**
10	0.585	0.173
20	1.812	0.465
50	13.346	6.028

study for the CDW and its similarity to the CDD. Thereafter, we present an $O(n^2)$ algorithm for a the distinct processing time case and an $O(n)$ algorithm for the unit processing time case, to optimize a given job sequence and prove the run-time complexity and its optimality with respect to the solution value. We applied our algorithms to the benchmark instances provided by Biskup and Feldmann [2]. In the future we intend to implement metaheuristic approaches using graphics processing units (GPUs) and provide speed-ups in runtime for all the instances.

ACKNOWLEDGEMENTS

The research project was promoted and funded by the European Union and the Free State of Saxony, Germany. The authors take the responsibility for the content of this publication.

REFERENCES

[1] Abhishek Awasthi, Jörg Lässig, Oliver Kramer, and Thomas Weise, "Common Due-Window Problem: Polynomial Algorithms for a Given Processing Sequence," *Proceedings of the IEEE Symposium on Computational Intelligence in Production and Logistics Systems (CIPLS'14), Proceedings of the IEEE Symposium Series on Computational Intelligence (SSCI'14)*, pages 32–39, December 9–12, 2014, Orlando, FL, USA: Caribe Royale All-Suite Hotel and Convention Center, Los Alamitos, CA, USA: IEEE Computer Society Press, http://www.it-weise.de/research/publications/ALKW2014CDWPPAFAGPS/ALKW2014CDWPPAFAGPS.pdf

[2] D. Biskup and M. Feldmann, "On scheduling around large restrictive common due windows," *European Journal of Operational Research*, vol. 162, no. 3, pp. 740 – 761, 2005.

[3] A. Seidmann, S. Panwalkar, and M. Smith, "Optimal assignment of due-dates for a single processor scheduling problem," *The International Journal Of Production Research*, vol. 19, no. 4, pp. 393–399, 1981.

[4] J. Kanet, "Minimizing the average deviation of job completion times about a common due date," *Naval Research Logistics Quarterly*, vol. 28, no. 4, pp. 643–651, 1981.

[5] S. Panwalkar, M. Smith, and A. Seidmann, "Common due date assignment to minimize total penalty for the one machine scheduling problem," *Operations Research*, vol. 30, no. 2, pp. 391–399, 1982.

[6] J. Hoogeveen and S. Van de Velde, "Scheduling around a small common due date," *European Journal of Operational Research*, vol. 55, no. 2, pp. 237–242, 1991.

[7] T. Cheng, "Optimal due-date assignment and sequencing in a single machine shop," *Applied Mathematics Letters*, vol. 2, no. 1, pp. 21–24, 1989.

[8] R. James, "Using tabu search to solve the common due date early/tardy machine scheduling problem," *Computers & Operations Research*, vol. 24, no. 3, pp. 199–208, 1997.

[9] F. Krämer and C. Lee, "Due window scheduling for parallel machines," *Mathematical and Computer Modelling*, vol. 20, no. 2, pp. 69 – 89, 1994.

[10] C. Lee, *Earliness-tardiness Scheduling Problems with Constant Size of Due Date Window*. Department of Industrial and Systems Engineering, University of Florida, 1991. [Online]. Available: http://books.google.de/books?id=ks0GHAAACAAJ

[11] S. Liman, S. Panwalkar, and S. Thongmee, "Determination of common due window location in a single machine scheduling problem," *European Journal of Operational Research*, vol. 93, no. 1, pp. 68 – 74, 1996.

[12] ——, "A single machine scheduling problem with common due window and controllable processing times," *Annals of Operations Research*, vol. 70, no. 0, pp. 145–154, 1997.

[13] Z. Chen and C. Lee, "Parallel machine scheduling with a common due window," *European Journal of Operational Research*, vol. 136, no. 3, pp. 512 – 527, 2002.

[14] G. Wan, "Single machine common due window scheduling with controllable job processing times," in *Combinatorial Optimization and Applications*, ser. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2007, vol. 4616.

[15] H. Zhao, H. Ma, G. Han, and L. Zhao, "A ptas for common due window scheduling with window penalty on identical machines," in *Computer Application and System Modeling (ICCASM), 2010 International Conference on*, vol. 10, Oct 2010, pp. 648–652.

[16] W. Yeung, T. Choi, and T. Cheng, "Optimal scheduling of a single-supplier single-manufacturer supply chain with common due windows," *IEEE Transactions on Automatic Control*, vol. 55, no. 12, pp. 2767–2777, 2010.

[17] T. Cheng, S. Yang, and D. Yang, "Common due-window assignment and scheduling of linear time-dependent deteriorating jobs and a deteriorating maintenance activity," *International Journal of Production Economics*, vol. 135, no. 1, pp. 154 – 161, 2012.

[18] E. Gerstl and G. Mosheiov, "Due-window assignment problems with unit-time jobs," *Applied Mathematics and Computation*, vol. 220, no. 0, pp. 487 – 495, 2013.

[19] Y. Yunqiang, T. Cheng, C. Hsu, and C. Wu, "Single-machine batch delivery scheduling with an assignable common due window," *Omega*, vol. 41, no. 2, pp. 216 – 225, 2013.

[20] A. Janiak, W. Janiak, M. Kovalyov, E. Kozan, and E. Pesch, "Parallel machine scheduling and common due window assignment with job independent earliness and tardiness costs," *Information Sciences*, vol. 224, pp. 109 – 117, 2013.

[21] A. Janiak, T. Kwiatkowski, and M. Lichtenstein, "Scheduling problems with a common due window assignment: A survey," *International Journal of Applied Mathematics and Computer Science*, vol. 23, 2013.

[22] N. Hall, W. Kubiak, and S. Sethi, "Earliness–tardiness scheduling problems, ii: deviation of completion times about a restrictive common due date," *Operations Research*, vol. 39, no. 5, pp. 847–856, 1991.

[23] A. Awasthi, J. Lässig, and O. Kramer, "Common due-date problem: Exact polynomial algorithms for a given job sequence," in *15th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC), 2013*, Sept 2013, pp. 258–264.

[24] A. Awasthi, O. Kramer, and J. Lässig, "Aircraft landing problem: An efficient algorithm for a given landing sequence," in *16th IEEE International Conferences on Computational Science and Engineering (CSE 2013)*, 2013, pp. 20–27.

[25] A. Awasthi, J. Lässig, and O. Kramer, *A Novel Approach to the Common Due-Date Problem on Single and Parallel Machines*, ser. Solving Computationally Extensive Engineering Problems: Methods and Applications. Springer, New York, Heidelberg, Berlin, to appear, 2014.

[26] P. Salamon, P. Sibani, and R. Frost, *Facts, Conjectures, and Improvements for Simulated Annealing*. Society for Industrial and Applied Mathematics, 2002.

[27] M. Gen, Y. Tsujimura, and E. Kubota, "Solving job-shop scheduling problems by genetic algorithm," in *IEEE International Conference on Systems, Man, and Cybernetics, 1994. Humans, Information and Technology.*, vol. 2, 1994, pp. 1577–1582.

[28] J. Kim, "Genetic algorithm stopping criteria for optimization of construction resource scheduling problems," *Construction Management and Economics*, vol. 31, no. 1, pp. 3–19, 2013.

[29] J. Lässig and D. Sudholt, "General upper bounds on the runtime of parallel evolutionary algorithms," 2013.

[30] J. Lässig and K. Hoffmann, "Threshold-selecting strategy for best possible ground state detection with genetic algorithms," *Phys. Rev. E*, vol. 79, p. 046702, April 2009.

This is a preview version of the paper [1] (see below for the reference). Read the full piece in the proceedings.


```
@inproceedings{ALKW2014CDWPPAFAGPS,
  author    = {Abhishek Awasthi and J{\"{o}}rg L{\"{a}}ssig and
               Oliver Kramer and Thomas Weise},
  title     = {Common Due-Window Problem: Polynomial Algorithms for
               a Given Processing Sequence},
  booktitle = {Proceedings of the IEEE Symposium on Computational
               Intelligence in Production and Logistics Systems
               (CIPLS'14), Proceedings of the  IEEE Symposium Series
               on Computational Intelligence (SSCI'14)},
  publisher = {Los Alamitos, CA, USA: IEEE Computer Society Press},
  address   = {Orlando, FL, USA: Caribe Royale All-Suite Hotel and
               Convention Center},
  year      = {2014},
  month     = dec # {~9--12, },
  pages     = {32--39},
  doi       = {10.1109/CIPLS.2014.7007158},
},
```