Management Science and Financial Engineering
Vol 18, No 1, May 2012, pp.1-12
ISSN 2287-2043 | EISSN 2287-2361 |

Common Due-Date Assignment and Scheduling with Sequence-Dependent Setup Times: a Case Study on a Paper Remanufacturing System

Jun-Gyu Kim
Department of Industrial Engineering, Hanyang University
Ji-Su Kim
Department of Industrial Engineering, Hanyang University
Dong-Ho Lee*
Department of Industrial Engineering/Graduate School of Technology and Innovation Management, Hanyang University

(Received: September 24, 2011 / Revised: December 10, 2011 / Accepted: March 16, 2012)

ABSTRACT

In this paper, we report a case study on the common due-date assignment and scheduling problem in a paper remanufacturing system that produces corrugated cardboards using collected waste papers for a given set of orders under the make-to-order (MTO) environment. Since the system produces corrugated cardboards in an integrated process and has sequence-dependent setups, the problem considered here can be regarded as common due-date assignment and sequencing on a single machine with sequence-dependent setup times. The objective is to minimize the sum of the penalties associated with due-date assignment, earliness, and tardiness. In the study, the earliness and tardiness penalties were obtained from inventory holding and backorder costs, respectively. To solve the problem, we adopted two types of algorithms: (a) branch and bound algorithm that gives the optimal solutions; and (b) heuristic algorithms. Computational experiments were done on the data generated from the case and the results show that both types of algorithms work well for the case data. In particular, the branch and bound algorithm gave the optimal solutions quickly. However, it is recommended to use the heuristic algorithms for large-sized instances, especially when the solution time is very critical.

Keywords: Due-Date Assignment, Scheduling, Paper Remanufacturing, Case Study

* Corresponding Author, E-mail: leman@hanyang.ac.kr

1. INTRODUCTION

Common due-date assignment and scheduling, which is the problem of determining the common due-date as well as the job schedule, has been received considerable attention during the last decades due to the just-in-time concept. In general, due-date assignment has a certain practical implication when a company offers due-dates of products to its customers during sale negotiations or offers a price reduction when the due-date is far away from the expected one. The earlier the due-dates are set, the higher the probability of the loss of customer goodwill since the products may not be completed or delivered on time. On the other hand, the later the due-dates are set, the higher the probability of having high inventory due to the early completions of products. In fact, there are various practical situations where due-dates are negotiated rather than simply set by customers.

This paper reports a case study on common due-date assignment and scheduling in the D paper remanufacturing system, located in Ansan, South Korea. Remanufacturing, the most advanced product recovery option, processes used or end-of-life products in such a way that their qualities are as good as new in terms of appearance, reliability, and performance (Lund, 1984). Using waste papers collected by the third party logistics companies, the D company produces corrugated cardboards of different grades in the form of large rolls and then they are cut into different sizes depending on customer orders. In general, the corrugated cardboards are used to make packaging boxes, etc.

In the paper remanufacturing system, the sequence-dependent setups must be considered since different types of raw materials are used according to product types. In other words, it is needed to clean up the equipment after processing a job (in the form of batch) if the job just completed is different from the job to be processed. Also, the system produces the corrugated cardboards in an integrated process. In other words, although the corrugated cardboards are produced by a series of operations, the entire system can be considered as a single machine. Note that paper manufacturing is generally classified as the process industry. The details of the manufacturing process are given in Section 2. Since the company produces corrugated cardboards in an integrated process and has sequence-dependent setup times, the problem considered here can be regarded as common due-date assignment and sequencing on a single machine with sequence-dependent setup times.

There are a few previous research articles on production planning and scheduling in paper manufacturing systems. Gupta and Magnusson (Gupta and Magnusson, 2005) consider the lot sizing and scheduling problem for sandpaper manufacturing system with sequence-dependent setups, and suggest a heuristic algorithm after formulating the problem as the basic capacitated lot sizing and scheduling problem (CLSP) where the setup condition at the end of each period is maintained. Bouchriha and D'Ouhimmou (2007) report another case study on the lot sizing and scheduling problem for a paper manufacturing company in Canada. They consider the problem with sequence-dependent setup times while considering the common cycle, and suggest a mixed integer programming model. Also, Kim *et al*. (2008) suggest heuristic algorithms for the CLSP in a paper remanufacturing system.

Most of the case studies on production planning and scheduling in paper manufacturing systems are done under the make-to-stock (MTS) environment. Unlike these, we performed a case study under the situation that the system is operated under the make-to-order (MTO) environment. In fact, the paper remanufacturing company is under the situation that the system is being changed from the MTS to the MTO environment. Therefore, a set of orders are given from customers and hence finding efficient production schedules becomes one of important system operation problems.

There are many theoretical articles on the single machine common due-date assignment and sequencing problem. Panwalkar *et al*. (1982) consider the problem for the objective of minimizing the sum of the penalties associated with due-date assignment, earliness and tardiness, and suggest an optimal algorithm in which the common due-date is set in advance using a preliminary analysis and then each job is sequenced based on the weight value corresponding to each position. Later, Cheng (1986) proposes a linear programming model for the problem of Panwalkar *et al*. (1982). Various extensions of the basic problem can be found in the literature (Quaddus, 1987; Baker and Scudder, 1989; Chen, 1996; Biskup and Jahnke, 2001; Ng *et al*., 2003; Cheng *et al*., 2002; Kim and Lee, 2009). Other extensions, such as other objectives and machine environments, can be found in the literature (Birman and Mosheiov, 2004; Chen *et al*., 1997; Cheng, 1990; Cheng and Kovalyov, 1996; Cheng *et al*., 1996; Cheng *et al*., 2002; Diamond and Cheng, 2000; Dvir, 2008; Dvir and George, 2006; Hall, 1986; Li *et al*., 2008; Mosheiov, 2001; Xia *et al*., 2008). See Gordon *et al*. (2002) for a literature review on various common due-date assignment and scheduling problems, especially those on a single machine.

This paper reports a case study on common due-date assignment and sequencing in the D paper remanufacturing system. In fact, this paper is a variation of Kim *et al*. (2008) that considers the capacitated lot-sizing and scheduling problem (CLSP) with sequence-dependent setups under the MTS environment. The problem is to determine the lot sizes as well as the sequence of lots for the objective of minimizing the sum of setup and inventory holding costs while satisfying the demand and the machine capacity over a given planning horizon. On the other hand, this paper considers the common due-date assignment and sequencing problem under the MTO environment for a given set of jobs in each customer order, and the problem is to determine the common due-date of the customer order as well as the sequence of jobs. The objective is to minimize the sum of the penalties associated with due-date assignment, earliness, and tardiness. In this study, the earliness and tardiness penalties imply inventory holding and backorder costs, respectively. To solve the problem, we adopted two algorithms of Kim and Lee (2009): (a) optimal branch and bound algorithm; and (b) heuristics. To show the applicability and performances of the algorithms, computational experiments were done on the data generated from the case and the results are reported.

The rest of this paper is organized as follows. The next section describes the system and then the problem considered in this paper. Section 3 explains the solution algorithms and Section 4 reports the test results. Finally, Section 5 gives a summary and discussion.

2. SYSTEM AND PROBLEM DESCRIPTIONS

2.1 System description

The D paper remanufacturing system makes various types of corrugated cardboards using waste papers collected by the third party logistics companies. According to the raw materials and product quality, the products can be largely classified into four categories: (a) product B (best quality) that uses only pulp and imported waste papers; (b) product S that uses either white or colored pulp; (c) product C that uses both liner board and corrugating medium; and (d) product K that uses domestic waste paper. As stated earlier, all products are produced in the form of paper roll with 4000 cm, and they are cut into final products with different sizes depending on customer orders. In this study, we consider the four product types before the cutting operation. The facility is continuously operated for full time with three shifts in a day except for maintenance works.

The detailed remanufacturing process of the paper rolls can be described as Figure 1. First, in the dissociation process, raw materials in the forms of collected waste paper, pulp and others are dissolved and passed through the quarantine step that removes wastes. Second, the materials are concentrated and go through the breathability process that put into the water and beat to a pulp, which gives wet materials for manufacturing papers. Third, the wet materials are processed repeatedly through the compression and drying processes that produce dried paper forms. Finally, the products with the width of 4000 cm are produced in the form of roll after performing the polishing process.

The corrugated cardboard manufacturing system is a type of process industry, such as chemical, oil refining, etc., and hence the entire system can be regarded as a single machine. In other words, the products are standardized and the processes required for each product are closely connected and well balanced. Also, the system is currently being operated in the MTS environment. However, due to uncertain customer orders and short lead times, the system operation is being changed into the MTO environment. This results that the scheduling problem is more important than the production planning problem. Also, it is needed to clean up the equipment after processing a job if the job just completed is different from the job to be processed. Therefore, the sequence-dependent setups must be considered since different types of raw materials are used in the production process.

2.2 Problem description

The problem considered here can be regarded as single machine common due-date assignment and sequencing with sequence-dependent setup times. Here, due-date assignment is additionally considered since there is a conflict that customers wish to receive their products as soon as possible while the company wants to have enough production time due to short production capacity. Note that the common due-date, which may be

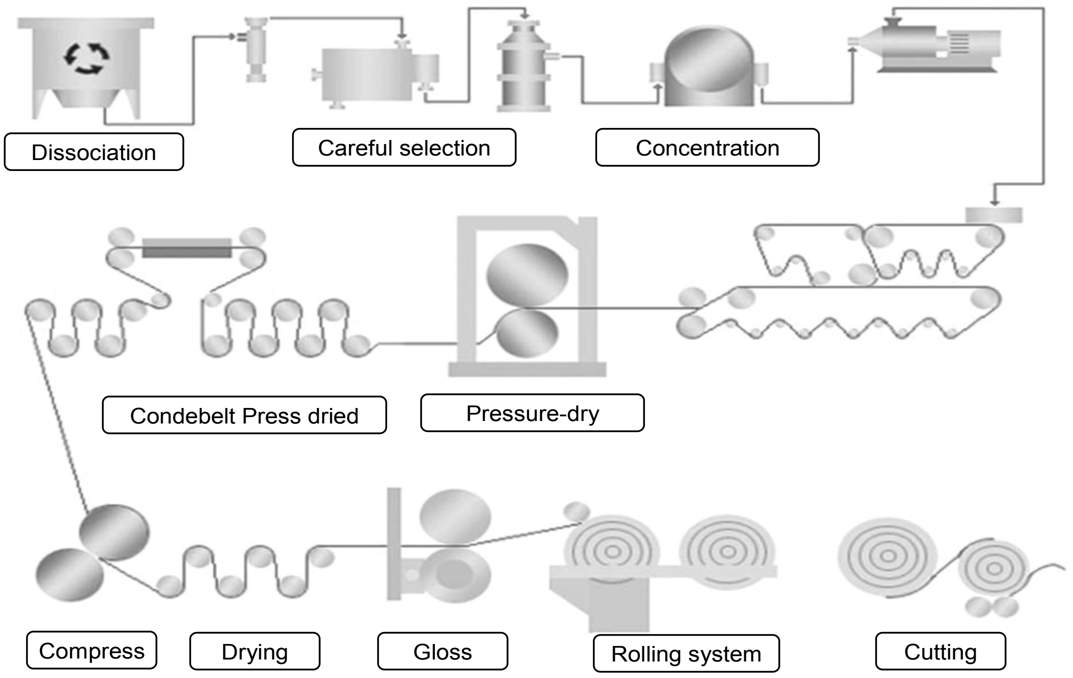

Figure 1. Remanufacturing Process of Corrugated Cardboards

different from the real due-date, is assigned to each customer order, where an order consists of one or more product types with specified quantities. Therefore, we solve the common due-date assignment and sequencing problem for each customer order. It is assumed that a set of customer orders to be produced in the upcoming period are given from the upper production planning decision.

Now, the problem considered in this study can be briefly described as: *for a given set of jobs in each customer order, we determine the common due-date as well as the job sequence while considering the sequence-dependent setup times for the objective of minimizing the sum of penalties associated with due-date assignment, earliness, and tardiness.* In this study, the penalties, associated with earliness and tardiness, are considered in the forms of inventory holding and backorder costs, respectively.

We consider a static version of the problem. That is, all jobs are available for processing simultaneously at the point of scheduling decision. Also, it is assumed that all job descriptors, such as processing times, setup times, penalties are deterministic and given in advance. Other assumptions made in this problem are summarized as follows: (a) the system cannot process two or more jobs simultaneously; (b) job splitting and preemptions are not permitted; and (c) idle times caused by machine breakdowns are not considered. Although we consider a static and deterministic version of the problem, it can be easily seen that the problem considered here is NP-hard because its special case is known to be NP-hard (Rabadi *et al.*, 2004).

To represent the problem more clearly, a mixed integer programming model of Kim and Lee (2009) is presented below. Before presenting the model, the notations are summarized below.

Parameters

p_i	processing time of job i ($i = 1, 2, \cdots, n$)
s_{ij}	setup time required between two consecutive jobs i and j
E_i	earliness of job i, i.e., $\max\{0, d-C_i\}$, where d is the common due-date (decision variable)
T_i	tardiness of job i, i.e., $\max\{0, C_i-d\}$
α	penalty associated with earliness
β	penalty associated with tardiness
γ	penalty associated with assigning the common due-date
C_i	completion time of job i
L	large number

Decision variables

d common due-date

$Y_{ij} = 1$ if job i is performed directly before job j, and 0 otherwise

Mixed integer programming model

$$\text{Minimize} \quad \sum_{i=1}^{n}(\alpha \cdot E_i + \beta \cdot T_i + \gamma \cdot d)$$

subject to

$$C_i - d = T_i - E_i \quad \text{for } i = 1, 2, \cdots, n \tag{1}$$

$$C_i - C_j + L \cdot Y_{ij} \geq p_i + s_{ji} \tag{2}$$

$$\text{for } i = 1, 2, \cdots, n \text{ and } j = i+1, \cdots, n$$

$$C_j - C_i + L \cdot (1 - Y_{ij}) \geq p_j + s_{ij} \tag{3}$$

$$\text{for } i = 1, 2, \cdots, n \text{ and } j = i+1, \cdots, n$$

$$C_i \geq p_i + s_{0i} \quad \text{for } i = 1, 2, \cdots, n \tag{4}$$

$$C_i,\ E_i,\ T_i,\ \geq 0 \quad \text{for } i = 1, 2, \cdots, n \tag{5}$$

$$d \geq 0 \tag{6}$$

$$Y_{ij} \in \{0, 1\} \tag{7}$$

$$\text{for } i = 1, 2, \cdots, n \text{ and } j = i+1, \cdots, n$$

The objective function denotes the sum of earliness, tardiness and due date assignment penalties that depend on the completion time of each job and the common-due-date. Constraint (1) specifies the amounts of earliness and tardiness while ensuring that T_i and E_i cannot be positive at the same time. Constraints (2) and (3) are used to represent the precedence relation between jobs i and j. Constraint (4) represents the minimum completion time of each job. That is, the completion time of an arbitrary job should be larger than or equal to the sum of its processing time and the initial setup time. Finally, the constraints (6) and (7) represent the conditions of the decision variables.

3. SOLUTION ALGORITHMS

This section explains the solution algorithms: (a) optimal branch and bound algorithm; and (d) heuristics. Before presenting the algorithms, we explain the method to set the common due-date.

3.1 Setting the Common Due-Date

As explained in Kim and Lee (2009), the following two propositions specify the optimal common due-date. Proposition 1 specifies that the common due-date must coincide with the completion time of a job in a given sequence. Also, by differentiating the objective function with respect to the common due-date and setting it equal to zero, we can specify the optimal common due-date. The details are given in Proposition 2. In the propositions, where α, β, and γ are the penalties associated with earliness, tardiness, and due-date assignment, respectively. Also, $[j]$ denotes the index for the job sequenced at the jth position, $j = 1, 2, \cdots, n$. See Appendix A for a numerical example for the two propositions.

Proposition 1: *For any specified sequence S, there exists an optimal value of common due-date d which coincides with the completion time of one of the jobs in S.*

Proposition 2: *For any specified sequence S, there exists an optimal common due-date equal to* $C_{[k]}$, *where k* *is the smallest integral value greater than or equal to* $n(\beta-\gamma)/(\alpha+\beta)$.

3.2 Branch and Bound (B&B) Algorithm

Before explaining the branching scheme, we calculate the positional weights according to Proposition 3. (See Kim and Lee (2009) for its proof.) Note that this result is an extension of the method to calculate the positional weights proposed by Panwalkar *et al.* (1982) by considering the sequence-dependent setup times. In this proposition, $AP_{ij} = s_{ij}+p_j$, where s_{ij} and p_j denote setup time required between two consecutive jobs i and j and processing time of the latter job j, respectively. Note that the job in the first position does not have the setup time, i.e., $AP_{[0][1]} = p_{[1]}$. See Appendix A for a numerical example for the proposition.

Proposition 3: *For any sequence, the objective function can be reformulated as*

$$w_1 \cdot p_{[1]} + \sum_{j=2}^{n} w_j \cdot AP_{[j-1][j]},$$

where $w_j = (j-1)\cdot\alpha + n\cdot\gamma$ *if* $j \le k$, *and* $\beta\cdot(n-j+1)$, *otherwise. Here,* w_j *is the positional weight when a job occupies the jth position and k is the index for the job that determines the optimal common due-date.*

Since the common due-date can be fixed using the Propositions 1 and 2, the branching scheme is explained with a B&B tree for a given common due-date. In the B&B tree, each node (except for the root) denotes a partial job sequence considering the positional weights given in Proposition 3, and each level corresponds to the number of jobs fixed in the partial sequence.

The branching starts at the top of the tree, i.e., level 0, where no jobs have been assigned to any position in the sequence. At level 1, n nodes are branched from the root node and the corresponding job is assigned to the position with the largest position weight. At level 2, n-1 nodes are branched from each of the nodes at level 1, and the jobs are assigned to the position with the second largest position weight, and so on. For node selection, the depth-first rule is used. That is, if the current node is not fathomed, the next node to be considered is its child node with the smallest job index. In this paper, the branching is done using the positional weights obtained from Proposition 3 since the job with the largest positional weight may contribute to minimize the objective value. Figure 2 shows an example of the B&B tree for an instance with 4 jobs in which the common due-date coincides with the completion time of the second job.

Now, we explain a lower bound, which can be calculated at each node of the B&B tree, and upper bound, which is calculated once at the root node of the B&B tree, i.e., an initial solution. Note that the upper bound is obtained using the better one of the two heuristics that will be explained later. Detailed methods to obtain the lower and upper bounds are given below. Let PS_l denote the set of jobs included in the partial sequence at node l.

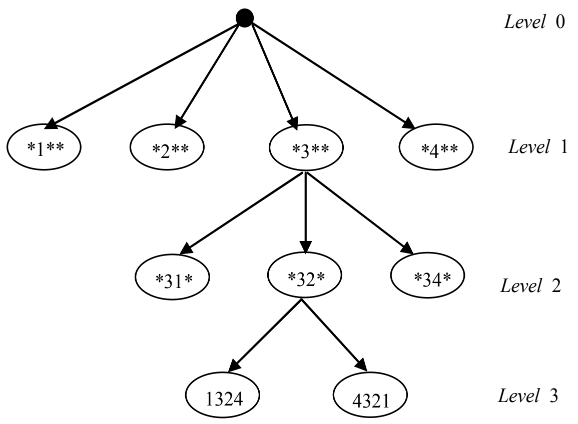

Figure 2. Branch and Bound Tree: Example

The lower bound, denoted by LB hereafter, is calculated as

$$AE + AT + PE + PT + DC,$$

where AE (AT) denotes the earliness (tardiness) penalty realized by the jobs in the partial sequence, and PE (PT) denotes the earliness (tardiness) penalty derived from the jobs not in PS_l. DC denotes the due-date assignment penalty realized by the jobs in the partial sequence. More formally, AE and AT can be represented as

$$AE = \alpha \cdot \sum_{j=k-e+2}^{k} (j-1)\cdot AP_{[j-1][j]} \quad \text{and}$$
$$AT = \beta \cdot \sum_{j=k}^{k+t-1} (n-j)\cdot AP_{[j][j+1]},$$

which is similar to the calculation method for the sequencing problem with earliness and tardiness penalties except that $AP_{[i][j]}$'s are used instead of the processing times. Here, e (t) denotes the number of jobs assigned before (after) the common due-date in the partial sequence. In addition, to derive PE and PT, we define

$$MAP_j = \min_{j=1,2,\cdots,n} \{AP_{ij}\}$$

for job $j \notin PS_l$ and $j \neq i$. Then, PE and PT are obtained by matching the job with the smallest value of MAP_j to the largest positional weight, the next smallest value of MAP_j to the next largest positional weight, and so on. More formally, PE and PT can be represented as

$$PE = \alpha \cdot \sum_{j=1}^{k-e+1} (j-1)MAP_{[j]} \quad \text{and}$$
$$PT = \beta \cdot \sum_{j=k+t}^{n} (n-j)MAP_{[j]}.$$

Finally, based on the sequence $[j]$ for $j = 1, 2, \cdots, n$, obtained from the above method, DC can be derived as

$$DC = \gamma \cdot n \cdot (P_{[1]} + \sum_{j=2}^{k-e+1} MAP_{[j]} + \sum_{j=k-e+2}^{k} AP_{[j-1][j]}).$$

Note that PE, PT and DC are valid elements for the lower bound since they are calculated using MAP_j for $j \notin PS_l$ instead of AP_{ij}.

The B&B algorithm incorporates a dominance property to reduce the search space. Note that the property given in the following proposition can be used in such a way that any node with the level less than or equal to four can be removed from further consideration if the condition given in the proposition holds. (See Kim and Lee (2009) for its proof.) See Appendix A for a numerical example for the proposition.

Proposition 4: *For four arbitrary consecutive jobs* $[r\text{-}1]$, $[r]$, $[r+1]$, *and* $[r+2]$ *in a partial sequence, the node with the smallest positional weight can be fathomed if*

$$w_{[r]}(AP_{[r-1][r+1]} - AP_{[r-1][r]}) + w_{[r+1]}(AP_{[r+1][r]} - AP_{[r][r+1]}) + w_{[r+2]}(AP_{[r][r+2]} - AP_{[r+1][r+2]}) < 0$$

where $w_{[r]}$ *is the weight of the rth position.*

3.3 Heuristic Algorithms

When the optimal B&B algorithm requires an excessive amount of computation time, the heuristic algorithms, called modified nearest neighborhood heuristic and clustering heuristic of Kim and Lee (2009), can be used. Each of the heuristics is explained below.

3.3.1 Modified nearest neighborhood heuristic

This heuristic consists of two phases: obtaining an initial solution and improvement. The initial solution is obtained with a method similar to the nearest neighbor heuristic for the traveling salesman problem (TSP) and then it is improved by iteratively interchanging the jobs in the current schedule.

First, a job is selected and assigned to the position with the largest positional weight. Then, if the position with the second largest positional weight is located before the common due-date, selected is the job j^* such that

$$j^* = \arg\min_{j \in U} \{BP_{ji}\},$$

where $BP_{ji} = s_{ji} + p_j$ and U denotes the set of unscheduled jobs. Otherwise, selected is the job j^* such that

$$j^* = \arg\min_{j \in U} \{AP_{ij}\},$$

where $AP_{ij} = s_{ij}+p_j$. Then, the selected job j^* is assigned to the corresponding position. This is done for the remaining positions in the non-increasing order of positional weights until a complete sequence is obtained. Since it is possible to obtain n different sequences, depending on the first job, the initial solution is set to the best one among the n sequences.

The detailed procedure to obtain the initial solution is summarized below. In the procedure, i_f denotes the index for the job to be positioned first.

Procedure 1: (Modified nearest neighborhood heuristic: obtaining an initial solution)
Step 1: Set $i_f = 1$.
Step 2: Initialize $U = \{1, 2, \cdots, n\}$, and do the following steps:
(a) Assign job i_f to the position with the largest positional weight and set $U = U \backslash \{i_f\}$.
(b) If the position with the second largest positional weight is located before the common due date, select job j^* with the minimum BP value, i.e.,

$$j^* = \arg\min_{j \in U} \{BP_{j,i_f}\},$$

where $BP_{ji} = s_{ji}+p_j$. Otherwise, select job j^* with the minimum AP value, i.e.,

$$j^* = \arg\min_{j \in U} \{AP_{i_f,j}\}$$

where $AP_{ij} = s_{ij}+p_j$.
(c) Assign the selected job j^* to the corresponding position, and set $U = U \backslash \{j^*\}$.
(d) Repeat steps (b) and (c) for the remaining positions in the non-increasing order of the positional weights until there is no remaining job.
Step 3: If the solution is improved, update the solution. Set $i_f = i_f + 1$. If $i_f > n$, stop. Otherwise go to Step 2.

To improve the initial solution, two methods for interchanging jobs, forward and backward interchanges, are used. In the forward interchange method, the jobs positioned before the common due-date are selected one by one according to the initial sequence and then the selected job is interchanged with those after the common due-date while considering the changes in the objective value. On the other hand, in the backward method carried out on the sequence obtained after the forward interchange method has terminated, the jobs positioned after the common due-date are selected one by one according to the sequence obtained from the forward interchange and then the selected jobs are interchanged with those before the common due-date. The detailed procedure for the improvement method is summarized below.

Procedure 2: (Modified nearest neighborhood heuristic: improvement)
Step 1: Set $r = 1$.

Step 2: (Forward interchange) Do the following steps:
(a) Set $t = i_d + 1$.
(b) Calculate the solution value after interchanging the jobs r and t in the current sequence.
(c) If this reduces the objective function value, update the solution.
(d) Set $t = t + 1$. If $t \le n$, go to Step 2(b). Otherwise, set $r = r + 1$ and go to Step 2(a).

Step 3: (Backward interchange) Do the following steps:
(a) Set $t = i_d + 1$ and $r = 1$.
(b) Calculate the solution value after interchanging the jobs r and t in the current sequence.
(c) If this reduces the objective function value, update the solution.
(d) Set $r = r + 1$. If $r \le i_d$, go to Step 3(b). Otherwise, set $t = t + 1$, $r = 1$ and go to Step 3(b).

3.3.2 Clustering heuristic

In this heuristic, an initial solution is obtained by solving the relaxed problems in which the sequence-dependent setup times are aggregated and added to the processing times, and it is improved by the forward and backward interchanging method explained earlier. The sequence-dependent setup times of each job are aggregated using two different methods: (a) the maximum sequence-dependent setup time; or (b) the average sequence-dependent setup time. More formally, the processing time of job j ($\neq i$) can be represented as either

$$p_j + \max_{i=1,2,\cdots,n}\{s_{ij}\} \text{ or } p_j + \frac{1}{n-1}\sum_{\substack{i=1\\ i\neq j}}^{n} s_{ij} .$$

To solve the relaxed problems, the optimal algorithm of Panwalkar *et al.* (1982) is used. Then, clusters are obtained by extracting the same partial sequences (before and after the common due-date) from the two sequences. Based on the clusters, a job is selected and assigned to the position with the largest positional weight explained earlier. Second, if the position with the second largest positional weight is located before (after) the common due-date, the job that directly precedes (succeeds) the initial job is selected if the initial job is included in the clusters positioned before (after) the common due-date. Otherwise, the job with the minimum *BP* (*AP*) value is selected. Then, the selected job is assigned to the position with the next largest positional weight. In this way, this heuristic constructs a complete sequence. As in the first heuristic, this heuristic sets the initial solution to the best one among the n sequences depending on the first job to be positioned. The detailed procedure to obtain the initial solution in the clustering heuristic is summarized below.

Procedure 3: (Clustering heuristic: obtaining an initial solution)

Step 1: Obtain the clustered jobs (after solving the two relaxed problems without sequence-dependent setup times) and set $i_f = 1$.

Step 2: Initialize $U = \{1, 2, \cdots, n\}$, and do the following steps:
(a) Assign job i_f to the position with the largest positional weight and set $U = U \backslash \{i_f\}$.
(b) If the position with the second largest positional weight is located before the common due date, select job j^* that directly precedes job i_f if job i_f is included in the clusters positioned before the common due date. Otherwise, select job j^* with the minimum *BP* value, i.e.,

$$j^* = \arg\min_{j\in U}\{BP_{j,i_f}\}$$

where $BP_{ji} = s_{ji} + p_j$.
(c) If the position with the second largest positional weight is located after the common due date, select job j^* that directly succeeds job i_f if job i_f is included in the clusters positioned after the common due date. Otherwise, select job j^* with the minimum *AP* value, i.e.,

$$j^* = \arg\min_{j\in U}\{AP_{i_f,j}\}$$

where $AP_{ij} = s_{ij} + p_j$.
(d) Assign the selected job j^* to the corresponding position, and set $U = U \backslash \{j^*\}$.
(e) Repeat steps (b), (c) and (d) for the remaining positions in the non-increasing order of the positional weights until there is no remaining job.

Step 3: If the solution is improved, update the solution. Set $i_f = i_f + 1$. If $i_f > n$, stop. Otherwise go to Step 2.

3.3.3 Numerical example

The two heuristic algorithms are explained more clearly using an example with 5 jobs. The processing times of jobs 1, 2, 3, 4 and 5 are 1, 2, 3, 4 and 5, respectively. The penalties associate with earliness, tardiness and assigning common due date are set to 2, 4, 1, respectively. ($\alpha = 2$, $\beta = 4$ and $\gamma = 1$) Also, the sequence dependent setup times are given below.

Table 1. Example: Sequence-Dependent Setup Times

Job	1	2	3	4	5
1	-	9	2	5	10
2	12	-	22	7	12
3	10	6	-	23	30
4	17	3	3	-	8
5	10	15	4	8	-

a) Setting the common due-date

Calculate the common due-date using Propositions 1 and 2, i.e.,

$$k = \lceil n \cdot (\beta - \gamma)/(\alpha + \beta) \rceil = \lceil 5 \cdot (4-1)/(2+4) \rceil = 3.$$

Then, the optimal common due-date becomes $C_{[3]}$.

b) *Calculating the positional weights*

$w_{[1]} = n \cdot \gamma = 1 \cdot 5 = 5$
$w_{[2]} = (j-1) \cdot \alpha + n \cdot \gamma = 1 \cdot 2 + 5 \cdot 1 = 7$
$w_{[3]} = 2 \cdot 2 + 5 \cdot 1 = 9$
$w_{[4]} = \beta \cdot (n - j + 1) = 4 \cdot (5 - 4 + 1) = 8$
$w_{[5]} = 4 \cdot (5 - 5 + 1) = 4$

c) *Obtaining the solutions*

1) *Modified nearest neighborhood heuristic*

Stage 1: Obtaining the initial solution

Step 1: Set $i_f = 1$

Step 2: Initialize $U = \{1, 2, 3, 4, 5\}$

- $i_f = 1$
  - Assign job 1 to the position with the largest positional weight (= $w_{[3]}$).
  - Partial sequence: (**1**) and $U = \{2, 3, 4, 5\}$
  - Select job j^* for the position with the second largest positional weight
    Calculate the minimum AP value since the position with the 2nd largest positional weight (= $w_{[4]}$) is located after the common due date.

  $j^* = argmin\{AP_{12}, AP_{13}, AP_{14}, AP_{15}\}$
  $= argmin\ \{s_{12}+p_2, s_{13}+p_3, s_{14}+p_4, s_{15}+p_5\}$
  $= argmin\ \{10, 5, 9, 15\} = 3$

  Partial sequence (**13*) and $U = \{2, 4, 5\}$
  - Select job j^* for the position with 3rd largest positional weight
    Calculate the minimum BP value since the position with the 3rd largest positional weight (= $w_{[2]}$) is located before the common due date.

    $j^* = argmin\{BP_{21}, BP_{41}, BP_{51}\}$
    $= argmin\ \{s_{21}+p_2, s_{41}+p_4, s_{51}+p_5\}$
    $= argmin\ \{14, 21, 15\} = 2$

    Partial sequence (*213*) and $U = \{4, 5\}$
  - In this way, we can obtain the final sequence (4, 2, 1, 3, 5) with the objective value of 285.
- $i_f = 2$
  - Partial sequence (**2**) and $U = \{1, 3, 4, 5\}$
  - Final sequence (1, 3, 2, 4, 5) with the objective value of 161
- $i_f = 3$
  - Partial sequence (**3**) and $U = \{1, 2, 4, 5\}$
  - Final sequence (5, 1, 3, 2, 4) with the objective value of 189
- $i_f = 4$
  - Partial sequence (**4**) and $U = \{1, 2, 3, 5\}$
  - Final sequence (3, 1, 4, 2, 5) with the objective value of 202
- $i_f = 5$
  - Partial sequence (**5**) and $U = \{1, 2, 3, 4\}$
  - Final sequence (2, 1, 5, 3, 4) with the objective value of 308

Step 3: Initial solution (1, 3, 2, 4, 5) (objective value = 161)

Stage 2: Improvement

Step 1: Forward interchange on the initial solution (1, 3, 2, 4, 5)

- Interchange jobs 1 and 4: (4, 3, 2, 1, 5) with objective value 231
- Interchange jobs 1 and 5: (5, 3, 2, 4, 1) with objective value 231
- Interchange jobs 3 and 4: (1, 4, 2, 3, 5) with objective value 363
- Interchange jobs 3 and 5: (1, 5, 2, 4, 3) with objective value 278
- Interchange jobs 2 and 4: (1, 3, 4, 2, 5) with objective value 298
- Interchange jobs 2 and 5: (1, 3, 5, 4, 2) with objective value 365

No improvement

Step 2: Backward interchange on the current solution (1, 3, 2, 4, 5)

- Interchange jobs 4 and 1: (4, 3, 2, 1, 5) with objective value 231
- Interchange jobs 4 and 3: (1, 4, 2, 3, 5) with objective value 363
- Interchange jobs 4 and 2: (1, 3, 4, 2, 5) with objective value 298
- Interchange jobs 5 and 1: (5, 3, 2, 4, 1) with objective value 231
- Interchange jobs 5 and 3: (1, 5, 2, 4, 3) with objective value 278
- Interchange jobs 5 and 2: (1, 3, 5, 4, 2) with objective value 365

No improvement

2) *Clustering heuristic*

Stage 1: Obtaining the initial solution

Step 1: Obtain the clustered jobs

- Job sequence using the maximum sequence-dependent setup time

  $p'_1 = p_1 + max\{s_{21}, s_{31}, s_{41}, s_{51}\}$
  $= 1 + max\{12, 11, 17, 10\} = 18$
  $p'_2 = p_2 + max\{s_{12}, s_{32}, s_{42}, s_{52}\}$
  $= 2 + max\{9, 6, 3, 15\} = 17$
  $p'_3 = p_3 + max\{s_{13}, s_{23}, s_{43}, s_{53}\}$
  $= 3 + max\{2, 22, 3, 4\} = 25$
  $p'_4 = p_4 + max\{s_{14}, s_{24}, s_{34}, s_{54}\}$
  $= 4 + max\{5, 7, 23, 8\} = 27$
  $p'_5 = p_5 + max\{s_{15}, s_{25}, s_{35}, s_{45}\}$
  $= 5 + max\{10, 12, 30, 8\} = 35$

  - Job sequence: 1, 3, 5, 4, 2 (Job 5$\rightarrow$ $w_{[3]}$

(= 9), Job 4→ $w_{[4]}$ (= 8), Job 3→ $w_{[2]}$ (= 7), Job 1→ $w_{[1]}$ (= 5), Job 2→ $w_{[5]}$ (= 4))

- Job sequence using the average sequence-dependent setup time
  $p'_1 = p_1 + average\{s_{21}, s_{31}, s_{41}, s_{51}\}$
  $= 1+(12+11+17+10)/4 = 13.5$
  $p'_2 = p_2 + average\{s_{12}, s_{32}, s_{42}, s_{52}\}$
  $= 2+(9+6+3+15)/4 = 10.25$
  $p'_3 = p_3 + average\{s_{13}, s_{23}, s_{43}, s_{53}\}$
  $= 3+(2+22+3+4)/4 = 10.75$
  $p'_4 = p_4 + average\{s_{14}, s_{24}, s_{34}, s_{54}\}$
  $= 4+(5+7+23+8)/4 = 14.75$
  $p'_5 = p_5 + average\{s_{15}, s_{25}, s_{35}, s_{45}\}$
  $= 5+(10+12+30+8)/4 = 25$
  - Job sequence: 3, 1, 5, 4, 2 (Job 5→ $w_{[3]}$(= 9), Job 4→ $w_{[4]}$(= 8), Job 1→ $w_{[2]}$(= 7), Job 3→ $w_{[1]}$(= 5), Job 2→ $w_{[5]}$(= 4))
- Job cluster: (5 – 4 – 2) after the common due date

Step 2: Initialize $U = \{1, 2, 3, 4, 5\}$ and set $i_f = 1$.

- $i_f = 1$
  - Assign job 1 to the position with the largest positional weight (= $w_{[3]}$).
  - Partial sequence: (**1**) and $U = \{2, 3, 4, 5\}$
  - Select job j^* for the position with the second largest positional weight
    Job i_f (= 1) is not included in the job cluster
    $j^* = argmin\ \{AP_{12}, AP_{13}, AP_{14}, AP_{15}\}$
    $= argmin\ \{10, 5, 9, 15\} = 3$
    Partial sequence (**13*) and $U = \{2, 4, 5\}$
  - Select job $j^*$for the position with 3rd largest positional weight
    Job i_f (= 1) is not included in the job cluster
    $j^* = argmin\{BP_{21}, BP_{41}, BP_{51}\}$
    $= argmin\ \{14, 21, 15\} = 2$
    Partial sequence (*213*) and $U = \{4, 5\}$
  - In this way, we can obtain the final sequence (4, 2, 1, 3, 5) with the objective value of 285.
- $i_f = 2$: final sequence (1, 3, 2, 4, 5) with the objective value of 161
- $i_f = 3$: final sequence (5, 1, 3, 2, 4) with the objective value of 189
- $i_f = 4$: final sequence (3, 1, 4, 2, 5) with the objective value of 202
- $i_f = 5$
  - Partial sequence: (**5**) and $U = \{1, 2, 3, 4\}$
  - Select job j^* for the position with the second largest positional weight
    Job i_f (= 5) is included in the job cluster after the common due date and job 4 directly succeeds job 5 in job cluster
    Partial sequence (**54*) and $U = \{1, 2, 3\}$
  - In this way, we can obtain the final sequence (3, 1, 5, 4, 2) with the objective value of 251

Step 3: Obtain the initial solution (1, 3, 2, 4, 5) (objective value = 161)

Stage 2: Improvement
Same as the modified nearest neighbour heuristic

4. TEST RESULTS

This section reports the test results on the case study. Since the system is currently operated under the make-to-stock environment, it is not possible to compare the performance of the algorithms with the existing method used in the company. Instead, we report the performances of the two types of algorithms: the optimal B&B algorithm and the heuristics. The performance measures used are: (a) CPU seconds for the optimal B&B algorithm; and (b) the percentage gaps from the optimal solution values for the two heuristics. Note that the CPU seconds of the heuristics are not reported here since they are less than 0.01 seconds. The B&B algorithm and the two heuristic algorithms were coded in C and the test was performed on a workstation with an Intel Xeon processor operating at 3.20GHz 120MHz clock speed.

To obtain the data required for the case study, we gathered the real demand data for a randomly selected planning period of six days. Table 2(a) shows the set of orders, together with product types and production quantities. In the test data, the number of jobs included in each order ranges from 6 to 12, and hence we can obtain the optimal solution using the B&B algorithm. Note that Kim and Lee (2009) report that the B&B algorithm can give optimal solutions for the problems up to 16 jobs (In this paper, the B&B algorithm is tested on the instances with 10, 12, 14, 15, 16 jobs). The process times of products B, S, C and K are 7, 6, 7 and 8.5 hours, respectively. Also, the sequence-dependent setup times are summarized in Table 2(b).

Since the company does not have the data for inventory holding and backorder costs, we generated the case instances with different penalty values. More specifically, 90 instances were generated, i.e., 10 instances for 9 combinations for three levels of inventory holding costs (10~20%, 20~30%, and 30~40% of the average price of the four product types) and five levels of backorder costs (20~30%, 30~40%, 40~50%, 50~60% and 60~70% of the average price of the four product types). In fact, we heard from the operation manager of the company that the inventory holding and the backorder costs do not exceed 50% and 80% of the product price, respectively. Here, inventory holding and backorder costs were estimated using the average price of the four product types. Also, the costs associated with the due-date assignment were generated as the backorder cost subtracted by 5%~10% of the average price of the four product types. Although we could not use the exact values of the three penalty values, we tried to increase the reliability of the results by testing the instances with various penalty values.

Table 2. Problem Data for the Case Instances

(a) Demand Requirements

Order number	Product types				Total number of jobs
	B	S	C	K	
1	2	2	3	1	8
2	5	1	4	2	12
3	3	1	2	4	10
4	3	1	3	4	11
5	3	1	3	5	12
6	3	1	2	4	10

(b) Sequence-Dependent Setup Times

Product types	Product types			
	B	S	C	K
B	-	1*	4.7	3
S	1	-	1.7	1.8
C	2.5	2.1	-	1
K	1.1	1.3	1	-

Note: * Sequence-dependent setup time (hr) from product B to S.

The test results are summarized in Table 3(a) and (b) that show the CPU seconds of the optimal B&B algorithm and the percentage gaps of the two heuristics, respectively. First, the optimal B&B algorithm required much short computation times since it solved only small-sized instances. However, we can easily see that the optimal B&B algorithm will not work well as the problem size increases. In this case, one can use the heuristic algorithms. As can be seen in Table 3(b), the two heuristic algorithms gave solutions within percentage gap of 5% for the case instances.

In fact, the percentage gaps of the modified nearest neighborhood heuristic (clustering heuristic) range from 2.66% (3.21%) to 4.36% (4.72%). Of the two heuristics, the modified nearest neighborhood heuristic was slightly better than the clustering heuristic in overall average gap. However, the result of the paired *t*-test shows that the two heuristics were not statistically different at a significant level of 0.01. In fact, the overall average percentage gaps of the modified nearest neighborhood heuristic and the clustering heuristic were 3.61% and 3.95%, respectively.

In summary, both types of algorithms work well for the case data. In other words, the B&B algorithm gave the optimal solutions very quickly since the number of jobs included in each order ranges only from 6 to 12. However, as reported in Kim and Lee (2009), the B&B algorithm cannot give the optimal solutions for the instances with more than 20 jobs, and hence the heuristic algorithms are more appropriate for large-sized instantces, especially when the solution is very critical.

Table 3. Test Results

(a) Branch and Bound Algorithm

Range of inventory costs	Range of backorder costs	CPU Seconds
10~20%	20~30%	0.03*
	30~40%	0.05
	40~50%	0.03
20~30%	30~40%	0.06
	40~50%	0.05
	50~60%	0.05
30~40%	40~50%	0.03
	50~60%	0.05
	60~70%	0.06
Average		0.05

Note: * Average CPU second out of 10 test instances.

(b) Heuristic Algorithms

Range of inventory costs	Range of backorder costs	Modified nearest neighborhood heuristic	Clustering heuristic
10~20%	20~30%	4.36*	4.72
	30~40%	3.25	3.53
	40~50%	3.27	3.75
20~30%	30~40%	3.26	3.56
	40~50%	4.00	4.20
	50~60%	4.35	4.67
30~40%	40~50%	4.20	4.36
	50~60%	2.66	3.21
	60~70%	3.13	3.54
Average		3.61	3.95

Notes: * Average percentage gap from the optimal solution values out of 10 test instances.

5. CONCLUDING REMARKS

In this paper, we reported a case study on the common due-date assignment and sequencing problem in a paper remanufacturing system that produces various corrugated cardboards using collected waste papers under the make-to-order environment. For a given set of orders, therefore, the problem is to determine the common due-date as well as the job sequence on a single machine for the objective of minimizing the sum of the penalties associated with due-date assignment, earliness, and tardiness. According to the characteristics of the system, in particular, we considered the sequence-dependent setup times. To solve the problem, we adopted two types of algorithms: (a) optimal branch and bound algorithm; and (b) heuristics. Computational experiments were done on the data generated from the case, and the results showed that both types of algorithms work well for the case instances. Although the branch and bound algorithm gave the optimal solutions quickly, it is recommended to use the heuristic algorithms for large-sized instances, especially when the solution time is critical.

For further research, it is needed to combine the scheduling problem with the upper production planning problem, which gives a core method for the integrated planning and scheduling system of the paper remanufacturing company. Also, the meta-heuristics, such as simulated annealing, tabu search, and genetic algorithm, can be used to obtain better solutions for large-size instances.

REFERENCES

Baker, K. R. and G. D. Scudder, "On the assignment of optimal due-dates," *Journal of the Operational Research Society* 40 (1989), 93-95.

Birman, M. and G. Mosheiov, "A note on due-date assignment in a two-machine flow-shop," *Computers and Operations Research* 31 (2004), 473-480.

Biskup, D. and H. Jahnke, "Common due-date assignment for scheduling on a single machine with jointly reducible processing times," *International Journal of Production Economics* 69 (2001), 317-322.

Bouchriha, H. M., and D'Ouhimmou, S., "Amours, "Lot sizing problem on a paper machine under a cyclic production approach," *International Journal of Production Economics* 105 (2007), 318-328.

Chen, D. W., S. Li, and G. C. Tang, "Single machine scheduling with common due date assignment in a group technology environment," *Mathematical and Computer Modelling* 25 (1997), 81-90.

Chen, Z. L., "Scheduling and common due date assignment with earliness-tardiness penalties and batch delivery costs," *European Journal of Operational Research* 93 (1996), 49-60.

Cheng, T. C. E., "A note on the common due-date assignment problem," *Journal of the Operational Research Society* 37 (1986), 1089-1091.

Cheng, T. C. E., "Common due-date assignment and scheduling for a single processor to minimize the number of tardy jobs," *Engineering Optimization* 16 (1990), 129-136.

Cheng, T. C. E. and M. Y. Kovalyov, "Bath scheduling and common due-date assignment on a single machine," *Discrete Applied Mathematics* 70 (1996), 231-245.

Cheng, T. C. E., C. Oguz, and X. D. Qi, "Due-date assignment and single machine scheduling with compressible processing times," *International Journal of Production Economics* 43 (1996), 29-35.

Cheng, T. C. E., Z. L. Chen, and N. V. Shakhlevich, "Common due-date assignment and scheduling with ready times," *Computers and Operations Research* 29 (2002), 1957-1967.

De, P., J. B. Ghosh, and C. E. Wells, "On the multiple-machine extension to a common due-date assignment and scheduling problem," *Journal of the Operational Research Society* 42 (1991), 419-422.

Diamond, J. E. and T. C. E. Cheng, "Error bound for common due date assignment and job scheduling on parallel machines," *IIE Transactions* 32 (2000), 445-448.

Dvir, S. and S. George, "Two due date assignment problems in scheduling a single machine," *Operations Research Letters* 34 (2006), 683-691.

Dvir, S., "Due date assignments and scheduling a single machine with a general earliness and tardiness cost function," *Computers and Operations Research* 35 (2008), 1539-1545.

Gordon, V., J. M. Proth, and C. Chu, "A survey of the state-of-the-art of the common due-date assignment and scheduling research," *European Journal of Operational Research* 139 (2002), 1-25.

Gupta, D. and T. Magnusson, "The capacitated lot-sizing scheduling problem with sequenced-dependent setup costs and setup times," *Computers and Operations Research* 32 (2005), 727-747.

Hall, N. G., "Scheduling problems with generalized due dates," *IIE Transactions* 18 (1986), 220-222.

Kim, H.-C., J.-G. Kim, and D.-H. Lee, "A case study on capacitated lot-sizing and scheduling in a paper remanufacturing system," Technical Report, Department of Industrial Engineering, Hanyang University, Seoul, South Korea (2008).

Kim, J.-G. and D.-H. Lee, "Algorithms for common due-date assignment and sequencing in a single machine with sequence-dependent setup times," *Journal of the Operational Research Society* 60 (2009), 1264-1272.

Li, C.-L., G. Mosheiov, and U. Yovel, "An efficient algorithm for minimizing earliness, tardiness, and due-date costs for equal-sized jobs," *Computers and Operations Research* 35 (2008), 3612-3619.

Lund, R. T., "Remanufacturing," *Technology Review* 87 (1984), 18-28.

Mosheiov, G., "A common due-date assignment problem on parallel identical machines," *Computers and Operations Research* 28 (2001), 719-732.

Ng, C. T. D., T. C. E. Cheng., M. Y. Kovalyov, and S. S. Lam, "Single machine scheduling with a variable common due date and resource-dependent processing times," *Computers and Operations Research* 30 (2003), 1173-1185.

Panwalkar, S. S., M. L. Smith, and A. Seidmann, "Common due-date assignment to minimize total penalty for the one machine scheduling problem," *Operations Research* 30 (1982), 391-399.

Quaddus, M. A., "A generalized model of optimal due-date assignment by linear programming," *Journal of the Operational Research Society* 38 (1987), 353-359.

Rabadi, G., M. Mollaghasemi, and G. C. Anagnostopoulos, "A branch-and-bound algorithm for the early/tardy machine scheduling problem with a common due-date and sequence-dependent setup time," *Computers and Operations Research* 31 (2004), 1727-1751.

Xia, Y., B. Chen, and J. Yue, "Job sequencing and due date assignment in a single machine shop with uncertain processing times," *European Journal of Operational Research* 184 (2008), 63-75.

Appendix A: AN EXAMPLE FOR PROPOSITIONS 1, 2, 3 AND 4

Consider a partial sequence such as (*3124*) with 6 jobs, where processing times of jobs 1, 2, 3, 4, 5 and 6 are 1, 2, 3, 4, 5 and 6, respectively. The sequence dependent setup times are $s_{12} = 3$, $s_{21} = 5$, $s_{31} = 13$, $s_{24} = 1$, $s_{14} = 3$, and $s_{32} = 8$. Also, the penalties associate with assigning common due-date, earliness and tardiness are set to 2, 5 and 7, respectively ($\alpha = 5$, $\beta = 7$ and $\gamma = 2$).

- Propositions 1 and 2: setting the optimal common due-date
  Optimal common due-date = $C_{[k]}$, where $k = \lceil n \cdot (\beta-\gamma)/(\alpha+\beta) \rceil = \lceil 6 \cdot (7-2)/(5+7) \rceil = 3$.
- Proposition 3: calculate the positional weights
  $w_{[1]} = n \cdot \gamma = 2 \cdot 6 = 12$
  $w_{[2]} = (j-1) \cdot \alpha + n \cdot \gamma = (2-1) \cdot 5 + 2 \cdot 6 = 17$
  $w_{[3]} = (3-1) \cdot 5 + 2 \cdot 6 = 22$
  $w_{[4]} = \beta \cdot (n-j+1) = 7 \cdot (6-4+1) = 21$
  $w_{[5]} = 7 \cdot (6-5+1) = 14$
  $w_{[6]} = 7 \cdot (6-6+1) = 7$

- Proposition 4: fathoming unnecessary solutions
  - Objective value for the current partial sequence (*3124*)
  $f(S) = w_{[r]} \cdot AP_{[r-1][r]} + w_{[r+1]} \cdot AP_{[r][r+1]}$
  $+ w_{[r+2]} \cdot AP_{[r+1][r+2]}$
  $= w_{[3]} \cdot AP_{[2][3]} + w_{[4]} \cdot AP_{[3][4]} + w_{[5]} \cdot AP_{[4][5]}$
  $= (2 \cdot \alpha + n \cdot \gamma) \cdot AP_{31} + 3 \cdot \beta \cdot AP_{12} + 2 \cdot \beta \cdot AP_{24}$
  $= (2 \cdot 5 + 2 \cdot 6) \cdot 14 + 3 \cdot 7 \cdot 5 + 2 \cdot 7 \cdot 5 = 473$
  - Objective value for partial sequence (*3214*) after interchanging the jobs 1 [r] and 2 [$r + 1$]
  $f(S') = w_{[r]} \cdot AP_{[r-1][r]} + w_{[r+1]} \cdot AP_{[r][r+1]}$
  $+ w_{[r+2]} \cdot AP_{[r+1][r+2]}$
  $= w_{[3]} \cdot AP_{[2][3]} + w_{[4]} \cdot AP_{[3][4]} + w_{[5]} \cdot AP_{[4][5]}$
  $= (2 \cdot \alpha + n \cdot \gamma) \cdot AP_{32} + 3 \cdot \beta \cdot AP_{21} + 2 \cdot \beta \cdot AP_{14}$
  $= (2 \cdot 5 + 2 \cdot 6) \cdot 10 + 3 \cdot 7 \cdot 6 + 2 \cdot 7 \cdot 7 = 444$

  Then, $f(S')$-$f(S)$ = 444-473 < 0, and hence the node with the smallest positional weight, that is, job 4 in position [r+2] can be removed from further consideration.