Available online at www.sciencedirect.com
ScienceDirect
European Journal of Operational Research 177 (2007) 325–343
EUROPEAN JOURNAL OF OPERATIONAL RESEARCH
www.elsevier.com/locate/ejor

Production, Manufacturing and Logistics

Considering manufacturing cost and scheduling performance on a CNC turning machine

Sinan Gurel, M. Selim Akturk *

Department of Industrial Engineering, Bilkent University, 06800 Bilkent, Ankara, Turkey

Received 20 March 2005; accepted 21 November 2005
Available online 23 February 2006

Abstract

A well known industry application that allows controllable processing times is the manufacturing operations on CNC machines. For each turning operation as an example, there is a nonlinear relationship between the manufacturing cost and its required processing time on a CNC turning machine. If we consider total manufacturing cost (F_1) and total weighted completion time (F_2) objectives simultaneously on a single CNC machine, making appropriate processing time decisions is as critical as making job sequencing decisions. We first give an effective model for the problem of minimizing F_1 subject to a given F_2 level. We deduce some optimality properties for this problem. Based on these properties, we propose a heuristic algorithm to generate an approximate set of efficient solutions. Our computational results indicate that the proposed algorithm performs better than the GAMS/MINOS commercial solver both in terms of solution quality and computational requirements such that the average CPU time is only 8% of the time required by the GAMS/MINOS.

Keywords: Scheduling; Bicriteria optimization; Controllable processing times; CNC machines; Manufacturing cost; Total weighted completion time

1. Introduction

Most of the scheduling literature assume fixed processing times. However, there are many industry applications where we can control the processing times. The best known example is the turning operation on a CNC machine. On a CNC turning machine, we can control the processing time of an operation by controlling the machining parameters. Since the scheduling problems are sensitive to the processing time data, we need appropriate processing time decisions to improve the scheduling objectives. When we consider a regular scheduling objective, we set processing time of each job as small as possible and then solve the scheduling problem. However, to achieve shorter processing times we have to use more resource. In a turning operation this resource is the cutting tool. As we decrease the processing time of an operation we incur more tooling cost.

* Corresponding author. Tel.: +90 312 290 1360; fax: +90 312 266 4054.
E-mail address: akturk@bilkent.edu.tr (M.S. Akturk).

0377-2217/$ - see front matter
doi:10.1016/j.ejor.2005.11.029

This study considers the situation where both total weighted completion time and cost performance are under consideration for a CNC turning machine. In order to find a set of efficient solutions for this bicriteria problem, we first present a mathematical model and derive optimality properties. Then, by utilizing these properties, we propose a new heuristic method to generate a set of approximate efficient solutions. Our results show that by integrating the machine scheduling and process planning decisions, we can generate a set of alternative solutions for the decision maker so that significant time/cost gains can be achieved.

On a CNC turning machine, increasing the cutting speed and/or feed rate decreases the processing time of an operation whereas it increases the tooling cost. Machining parameters selection problem dealing with this tradeoff is a well known problem in the literature. There are many studies in the literature that consider different objectives and solution methods. Malakooti and Deviprasad (1989) formulated machine parameter selection problem as a multiple objective decision making problem. They considered minimizing cost per part, minimizing production time and minimizing surface roughness objectives. Gopalakrishnan and Al-Khayyal (1991) provided a geometric programming based method to minimize machining and tooling costs. Choi and Bricker (1996) discussed the effectiveness of a geometric programming model in machining optimization problems. Lamond and Sodhi (1997) considered the cutting speed selection and tool loading decisions on a single cutting machine so as to minimize total processing time. Akturk and Avci (1996) propose a solution procedure to make tool allocation and machining conditions selection decisions simultaneously. Kayan and Akturk (2005) provide a mechanism to determine upper and lower bounds for the processing time of a turning operation.

In the scheduling literature, controllable processing time issue has been receiving increasing attention in recent studies. They deal with minimizing the processing cost and a scheduling objective simultaneously. The processing cost of a job is usually defined to be the cost of compressing the processing time of it. The survey of Nowicki and Zdrzalka (1990) lists results for sequencing problems with controllable processing times up to 1990. A review on controllable processing times in multi-objective scheduling is included in the recent review by Hoogeveen (2005). The first study in the area is by Vickson (1980a) which considers the objective of minimizing the sum of total processing cost and total completion time on a single machine, and also shows that the problem is polynomially solvable. Chen et al. (1997) show that the discrete controllable case for the same problem is also polynomially solvable. Daniel Ng et al. (2003) additionally consider batching and controllable setup times for the same objective. Vickson (1980b) considers total weighted completion time case which is recently shown to be NP-hard by Wan et al. (2001). Janiak et al. (2005) provide an alternative NP-hardness proof for the same problem and propose a polynomial time approximation algorithm. Karabati and Kouvelis (1997) discuss simultaneous scheduling and optimal processing time decision problem to maximize the throughput for a multi-product, deterministic flow line operated under a cyclic scheduling approach. These studies assumed linear job compression costs. A nonlinear relationship is considered between processing times and production resource by Shabtay and Kaspi (2004). They study a single machine scheduling problem to minimize total weighted flow time subject to a limited resource. Kayan and Akturk (2005) consider the bicriteria problem of minimizing the makespan and total manufacturing cost simultaneously on a CNC turning machine and provide methods to determine an approximate efficient frontier for the problem.

In the literature, most of the research on scheduling has focused on problems with a single objective. However, in the real world, we usually face a number of objectives. Process planning decisions focus on how to minimize manufacturing cost, whereas in the scheduling decisions the main aim is to optimize a scheduling criterion. Usually these two decisions are made independently. Since there is a significant interaction between the schedule performance and cost, it would be more beneficial to propose a model that combines these decisions to minimize total manufacturing cost and total weighted completion time, simultaneously. We also give an alternative model for the total completion time case, and present optimality properties for both problems. In practise, the weights of cost and time objectives for the decision maker may vary over time. If the workload is heavy, scheduling related objectives become more important. If it is relatively light, the manufacturing cost is very important. Therefore, we proposed a new method to generate an approximate efficient solution set for this bicriteria problem.

In multi-objective optimization problems, approximation quality of the generated efficient set is important to the decision maker. In the literature, there are different approximation quality evaluation metrics developed. These metrics are useful for comparing different algorithms. Tuyttens et al. (2000) consider the classical

linear assignment problem with two objectives for which they employ a multi-objective simulated annealing method, and provide two metrics to compare the results with an exact efficient set. Wu and Azarm (2001) propose some quality evaluation measures to compare efficient sets generated by different multi-objective optimization methods. A review and discussion on existing metrics is available in Zitzler et al. (2003). In this paper, we will employ three different metrics to compare the approximation quality of our proposed methods.

In this study, we basically focus on CNC turning machines, so we have a well defined and realistic manufacturing cost function for each job. This manufacturing cost function is nonlinear and convex. In our analysis, we assume the case where the manufacturing cost function might be different for each job due to different operational and surface quality requirements, and its required cutting tool. However, all our results are applicable for the cases where there exists sublot of jobs which are identical. Although we specifically consider manufacturing cost function for the turning operation, our results apply to any problem with nonlinear convex processing cost functions. Furthermore, all the properties and methods that we derive below apply to the linear cost function case as well. In the literature, to the best of our knowledge, the linear cost function case was not considered either.

In the next section, the problem definition is given and a mathematical model to find an efficient solution for a given total weighted completion time level is provided. We give optimality properties for the model. Based on these properties, an efficient frontier approximation method is presented in Section 3. Then, in Section 4, an alternative model for the total completion time problem is discussed. In Section 5, a numerical example is presented. Finally, in Section 6 the computational results and the performances of the proposed methods are discussed.

The notation used throughout the paper is as follows:

Decision variables

p_i processing time of job i
X_{ij} binary variable to state if job i precedes job j in the sequence
v_i cutting speed for operation i (fpm)
f_i feed rate for operation i (ipr)
U_i usage rate of the required cutting tool to process operation i

Parameters

p_i^{l} processing time lower bound for job i
p_i^{u} processing time level that gives minimum manufacturing cost for job i
w_i weight of job i
$f_i(p_i)$ manufacturing cost function of processing time for job i
α, β, γ speed, feed, depth of cut exponents for the required cutting tool of job i
C_i^{TL} Taylor's tool life constant for the required cutting tool of job i
C_{o} operating cost of the CNC turning machine (\$/min)
C_{s}, g, h, l specific coefficients of the surface roughness constraint of job i
C_{m}, b, c, e specific coefficients and exponents of the machine power constraint
C_{t_i} cost of cutting tool required to process job i (\$/tool)
D_i diameter of the generated surface for job i (in)
d_i depth of cut for job i (in)
H maximum available machine power (hp)
L_i length of the generated surface for job i (in)
S_i maximum allowable surface roughness for job i (μin.)

2. Problem definition

We have N jobs to be processed, and each job corresponds to a metal cutting operation that will be performed by a given cutting tool on a single CNC turning machine. Each job differs in terms of its manufacturing properties such as diameter, length, depth of cut and maximum allowable surface roughness and its cutting

tool, and a positive weight which shows its importance relative to the other jobs. The CNC turning machine can process one job at a time. We have two objectives, minimizing the total manufacturing cost of jobs on a CNC turning machine and minimizing their total weighted completion time. Therefore, we have to determine a job sequence and the corresponding processing times simultaneously. In order to solve this bicriteria problem, we have to consider process planning and scheduling problems simultaneously. One way to integrate these two decision making problems is through the proper selection of job processing times. Assuming a single pass operation, the processing time of job i on a CNC turning machine can be calculated as follows:

$$p_i = \frac{\pi D_i L_i}{12 v_i f_i}.$$

On the other hand, the tool usage rate of a job, U_i, is simply the ratio of its processing time to the tool life. If we use extended Taylor's tool life equation, T_i, to describe the tool life then

$$U_i = \frac{p_i}{T_i} = \frac{(\pi D_i L_i)/(12 v_i f_i)}{C_i^{\mathrm{TL}}/(v^{\alpha} f^{\beta} d^{\gamma})}.$$

The optimum machining parameters of the cutting speed (v_i) and the feed rate (f_i) for each job can be found by solving machining conditions optimization problem subject to the tool life, surface roughness and machine power constraints as discussed in Appendix A. The most commonly used objective function for the manufacturing cost of job i is the sum of the operating and tooling costs. Operating cost of job i is the cost of running the machine for p_i. We assume that C_o is constant and independent of selected machining parameters. Tooling cost for job i is the cost of its tool usage U_i. We also assume that setup time and the tool change times are negligible. In Appendix A, we give the geometric programming model for the problem and the optimality properties proved by Akturk and Avci (1996) and Kayan and Akturk (2005). They showed that the surface roughness constraint must be tight at the optimal solution. Using this property, the manufacturing cost of job i is given below as a function of p_i:

$$f_i(p_i) = C_o p_i + C_{t_i} U_i = C_o p_i + C_{t_i} \frac{d_i^{\gamma}}{C_i^{\mathrm{TL}}} \left(\frac{\pi D_i L_i}{12}\right)^{\left(\frac{\alpha h-\beta g}{h-g}\right)} \left(\frac{C_s d_i^{1}}{S_i}\right)^{\left(\frac{\alpha-\beta}{h-g}\right)} p_i^{\left(\frac{(1-\alpha)h-(1-\beta)g}{h-g}\right)}.$$

Furthermore, Kayan and Akturk (2005) showed that the nonlinear manufacturing constraints that limit the allowable ranges of the processing times can be replaced by a linear bound of $p_i^{\mathrm{l}} \leqslant p_i \leqslant p_i^{\mathrm{u}}$ for each job i when there is a regular scheduling performance measure, such as makespan or total completion time. For the determination of p_i^{l} and p_i^{u} values, we refer to Kayan and Akturk (2005). A typical manufacturing cost function behavior for a job is given in Fig. 1. Since jobs have different manufacturing properties, they will have different nonlinear convex manufacturing cost functions and different bounds on their processing times.

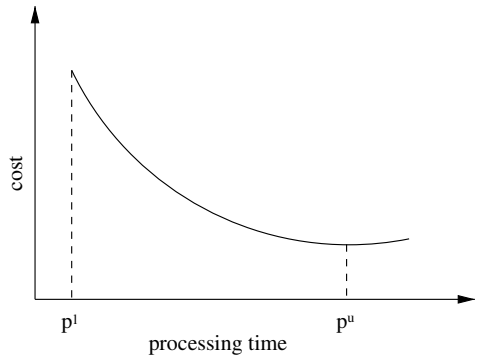

Fig. 1. A typical manufacturing cost function for a turning operation.

The mathematical model for the problem is as below:

$$\min\ F_1: \quad \sum_{i=1}^{N} f_i(p_i) = \sum_{i=1}^{N} C_o p_i + C_{t_i} U_i$$

$$\min\ F_2: \quad \sum_{j=1}^{N} \sum_{i=1}^{N} w_j p_i X_{ij} + \sum_{j=1}^{N} w_j p_j$$

$$\text{subject to} \quad X_{ij} + X_{ji} = 1, \quad i = 1, \ldots, N \text{ and } j = i+1, \ldots, N, \tag{1}$$

$$X_{ij} + X_{jk} + X_{ki} \geqslant 1, \quad j,k,l = 1, \ldots, N, \text{ and } j \neq k \neq l, \tag{2}$$

$$p_i^{\mathrm{l}} \leqslant p_i \leqslant p_i^{\mathrm{u}} \quad \forall i, \tag{3}$$

$$X_{ij} \in \{0,1\} \quad \forall i,j \text{ and } i \neq j. \tag{4}$$

In the mixed integer nonlinear programming (MINLP) model above, the first objective function (F_1) is the total manufacturing cost. The second objective function (F_2) is the total weighted completion time. Constraint set (1) is the precedence constraints to ensure that two jobs cannot precede each other at the same time. Constraint set (2) satisfies the triangular inequality among the jobs such that if job i precedes job j and job j precedes job k then job i precedes job k. We have constraint set (3) that sets the upper and lower bounds on the processing time of each job.

For the weighted completion time problem, the minimum value is attained when we set the processing times to their lower bounds, p_i^{l}. On the other hand, the manufacturing cost decreases when we increase the processing times, and the minimum manufacturing cost is attained at p_i^{u} for each job i. That means if we increase the processing time of a job, the manufacturing cost decreases but completion time of the job itself and all the following jobs increase. Therefore, we cannot minimize both objectives F_1 and F_2 at the same time, and hence the overall problem is to generate an efficient solution set for the decision maker. A solution $x\ (F_1(x), F_2(x))$ is said to be efficient with respect to the given bicriteria if there does not exist another solution $y\ (F_1(y), F_2(y))$ such that $F_1(y) \leqslant F_1(x)$ and $F_2(y) \leqslant F_2(x)$ with at least one holding as a strict inequality. The following lemma states that there are infinitely many efficient solutions for the problem.

Lemma 1.1. *The efficient solution set for the problem includes infinitely many points.*

Proof. This is due to the fact that the processing times of jobs are continuous and can take any value satisfying $p_i^{\mathrm{l}} \leqslant p_i \leqslant p_i^{\mathrm{u}}$. If we slightly decrease the processing time of any job i that will increase the total manufacturing cost (as shown in Fig. 1), but at the same time it will decrease the total weighted completion time. Hence, there are infinitely many possible F_2 (or F_1) levels for the problem and we can find infinitely many efficient solutions. Furthermore, efficient frontier can be represented as a continuous function on a (F_1, F_2) plot. □

In Fig. 2, an example for a set of efficient solutions is given. Solution Z_1 is the ideal solution for F_2 where $F_2(Z_1) = K^{\mathrm{l}}$ where the superscript l implies that K^{l} is achieved by setting $p_i = p_i^{\mathrm{l}}$ for each job i and applying the

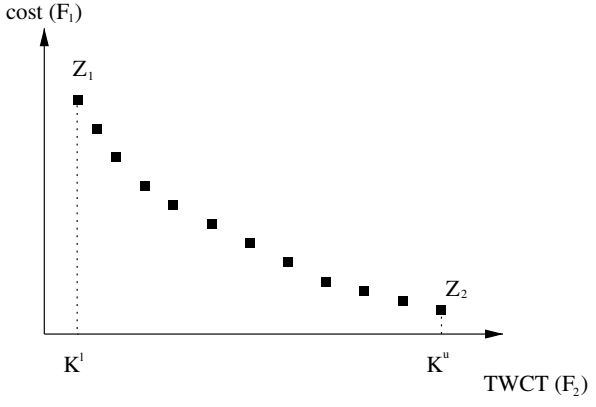

Fig. 2. An example set of efficient solutions.

weighted shortest processing time first (WSPT) rule by Smith (1956). According to the WSPT rule the jobs are ordered in the decreasing order of w_i/p_i to minimize total weighted completion time. At Z_1, total manufacturing cost is $F_1(Z_1) = \sum_{i=1}^{N} f_i(p_i^{\mathrm{l}})$. On the other hand, solution Z_2 is the ideal solution for F_1 where $F_1(Z_2) = \sum_{i=1}^{N} f_i(p_i^{\mathrm{u}})$ and it is achieved by setting $p_i = p_i^{\mathrm{u}}$ for each job i and jobs are ordered by the WSPT rule. At Z_2, total weighted completion time is $F_2(Z_2) = K^{\mathrm{u}}$, where the superscript u implies that the solution is achieved where all jobs are machined at processing time upper bounds.

In order to find a set of efficient solutions other than Z_1 and Z_2, we can consider F_2 as a constraint as in the formulation below and solve the resulting Single Criterion Problem (SCP) for different values of K.

$$\begin{aligned} \min\ F_1: \quad & \sum_{i=1}^{N} f_i(p_i) \\ \text{subject to} \quad & \sum_{j=1}^{N}\sum_{i=1}^{N} w_j p_i X_{ij} + \sum_{j=1}^{N} w_j p_j \leqslant K, \qquad (5) \\ & \text{and} \quad (1)\text{–}(4). \end{aligned}$$

Constraint (5) guarantees that the total weighted completion time (F_2) of the schedule is less than or equal to a predefined value K. We can solve this model by the MINLP solvers like GAMS/BARON (Brooke et al., 2004). To generate an efficient solution set of n points between Z_1 and Z_2 we can employ the following algorithm denoted as the SCP-based method:

Step 1. Calculate K^{u} and K^{l} by using the WSPT rule for p^{u} and p^{l} values, respectively.
Step 2. Set $\epsilon = (K^{\mathrm{u}} - K^{\mathrm{l}})/(n+1)$.
Step 3. For $k = 1$ to n, solve the SCP model for $K = K^{\mathrm{l}} + k\epsilon$.

The SCP-based method finds a set of efficient points by solving the SCP model iteratively, so we investigated the SCP model and found some useful properties for the problem.

Lemma 1.2. *When $K \leqslant K^{\mathrm{u}}$, the constraint (5) on F_2 must be tight at the optimal solution. This implies that the optimal schedule must satisfy the WSPT rule.*

Proof. When $K > K^{\mathrm{u}}$, total manufacturing cost can be minimized by setting all jobs to their minimum cost processing times (p^{u}) and ordering them by the WSPT rule. In this case, the constraint (5) can be loose at optimality. When $K \leqslant K^{\mathrm{u}}$, if constraint (5) is loose, then by increasing the processing time of a job, we can decrease the total manufacturing cost of the schedule. Therefore, a solution cannot be optimal if constraint (5) is loose. As a consequence of this result we can also state that the WSPT rule must be satisfied by an optimal schedule. Otherwise, we can reorder the jobs and find a better solution which violates the optimality. □

We proved that an optimal solution for the SCP must satisfy the constraint (5) as an equality and it must also satisfy the WSPT rule. The next property for the problem is about the non-cycling constraint set (2).

Lemma 1.3. *The non-cycling constraints, constraint set (2), are redundant for the SCP.*

Proof. Consider the SCP model without constraint set (2). We can easily see that Lemma 1.2 still holds for the new problem, otherwise we can improve the solution by resequencing the jobs and increasing the processing times. Lemma 1.2 states that the optimal solution must satisfy the WSPT rule which implies that the optimal solution cannot have cycles and always satisfies the constraint set (2). □

By using this result we can eliminate constraint set (2) when solving the SCP model in SCP-based method. Since constraint set (2) is defined for each job triple, removing it reduces most of the constraints in the model so that MINLP solvers can solve the problem more efficiently. From this point on, SCP will denote the single criterion problem without constraint set (2). Further, we consider the relaxation of the problem where the integrality assumption of X_{ij}'s is relaxed. Relaxation results a nonlinear programming (NLP) model for which we can state the following two properties.

Lemma 1.4. *For the relaxed SCP, in a local optimal solution, if $w_i/p_i > w_j/p_j$ for any job pair (i,j), then job i must precede job j, i.e. the solution must have $X_{ij} = 1$. This implies that a local optimal solution to the relaxed SCP must have binary X_{ij}'s.*

Proof. Suppose that there is a local optimal solution S which has non-integer X_{ij} values. We will show that by modifying S in a way to achieve an integer solution, we can improve F_2.

Consider a job pair (i,j) in S for which $w_i/p_i > w_j/p_j$ holds. Suppose that the precedence variables for the job pair (i,j) are as follows: $X_{ij} = \lambda$, $X_{ji} = 1 - \lambda$, where $0 \leqslant \lambda < 1$. Then, $\sum w_i C_i$ for S is as below:

$$F_2(S) = \Phi + w_i \times p_j \times (1 - \lambda) + w_j \times p_i \times \lambda,$$

where Φ is a constant value.

If we form a new solution S' from S by setting $X_{ij} = 1$ and $X_{ji} = 0$, we get

$$F_2(S') = \Phi + w_j \times p_i.$$

Obviously, $F_2(S') < F_2(S)$ and S' is a feasible solution to the SCP. This proves that a solution S with non-integer X_{ij} variables cannot be a local optimal solution. □

Considering Lemma 1.2 and the arguments above together, we conclude that in a local optimal solution to the relaxed problem all X_{ij} variables are binary. However, to be rigorous, we must also point out that in case $w_i/p_i = w_j/p_j$ for some job pair (i,j), we may have alternative optimal solutions where X_{ij} and X_{ji} are non-integer. This is because when $w_i/p_i = w_j/p_j$, $X_{ij} = \lambda$ and $X_{ji} = 1 - \lambda$, whatever value λ takes such that $0 \leqslant \lambda \leqslant 1$, $\sum w_i C_i$ calculated by the mathematical model remains the same. In practice, it is highly unlikely to face this situation since in our case p_i's are controllable variables that take real values.

Lemma 1.4 is an extremely important result to reduce the computational burden since we do not need MINLP solvers to solve the SCP. The problem can be solved by an NLP solver. However, NLP solvers can only guarantee to achieve local optimal solutions. From nonlinear programming theory we know that if the objective function of a problem is convex and the feasible region for the problem is a convex set, then a local optimum is a global optimum. Therefore, we investigated if the feasible region for the relaxed SCP is a convex set or not to see if NLP solvers could give us the global optimum.

Lemma 1.5. *The feasible region for the relaxed SCP is not a convex set, i.e. NLP solvers cannot guarantee global optimality for the problem.*

Proof. Consider two jobs i_1 and i_2 such that i_1 immediately precedes i_2, $X_{i_1 i_2} = 1$, in a solution called A_1. Let us suppose that $p_{i_1} = s_1$ and $p_{i_2} = s_2$ with weights w_1 and w_2, respectively. We also have $\frac{w_1}{s_1} > \frac{w_2}{s_2}$. $F_2(A_1) = Q + w_2 s_1 + w_1 s_1 + w_2 s_2 = K$, where Q is a constant. Next, consider another solution A_2 which is identical to A_1 except that i_1 and i_2 were pairwise interchanged and processing times were set to q_1 and q_2, respectively, and $\frac{w_2}{q_2} > \frac{w_1}{q_1}$ holds. $F_2(A_2) = Q + w_1 q_2 + w_1 q_1 + w_2 q_2 = K$.

Now consider a convex combination of A_1 and A_2 as the solution $A = \lambda A_1 + (1 - \lambda) A_2$. At A, $X_{i_1 i_2} = \lambda$, $X_{i_2 i_1} = 1 - \lambda$, $p_{i_1} = \lambda s_1 + (1 - \lambda) q_1$ and $p_{i_2} = \lambda s_2 + (1 - \lambda) q_2$. Then,

$$\begin{aligned}
F_2(A) &= Q + w_1 X_{i_2 i_1} p_{i_2} + w_2 X_{i_1 i_2} p_{i_1} + w_1 p_{i_1} + w_2 p_{i_2} \\
&= Q + (w_1(1 - \lambda) + w_2)(\lambda s_2 + (1 - \lambda) q_2) + (w_2 \lambda + w_1)(\lambda s_1 + (1 - \lambda) q_1) \\
&= Q + \lambda(s_1(w_1 + \lambda w_2) + s_2(w_2 + (1 - \lambda) w_1)) + (1 - \lambda)(q_1(w_1 + \lambda w_2) + q_2(w_2 + (1 - \lambda) w_1) \\
&= Q + \lambda(s_1 w_1 + s_2 w_2 + \lambda s_1 w_2 + (1 - \lambda) s_2 w_1) + (1 - \lambda)(q_1 w_1 + q_2 w_2 + \lambda q_1 w_2 + (1 - \lambda) q_2 w_1) \\
&> Q + \lambda(K - Q) + (1 - \lambda)(K - Q) > K
\end{aligned}$$

since $s_2 w_1 > s_1 w_2$ and $q_1 w_2 > q_2 w_1$. This shows that the feasible region for the relaxed SCP is not a convex set. □

Lemma 1.5 implies the potential existence of multiple local optimal solutions for the relaxed SCP. Although it does not prove NP-hardness of the SCP, we know that in general, computing a global minimum in a non-convex NLP is an NP-hard problem due to Murty and Kabadi (1987).

Lemma 1.5 is a direct consequence of the nonlinear terms p_iX_{ij} in constraint (5), so that the MINLP solvers may terminate with an integer local optimal solution. In order to find the global optimum for the SCP model, we propose the following linearized single criterion problem (LSCP) model below. In this model, we replace the constraint (5) with a set of linear constraints (6)–(10) and the term p_iX_{ij} is replaced with the variable Y_{ij}, where M is a large positive number.

$$\text{min } F_1: \quad \sum_{i=1}^{N} f_i(p_i)$$

$$\text{subject to} \quad \sum_{j=1}^{N}\sum_{i=1}^{N} w_jY_{ij} + \sum_{j=1}^{N} w_jp_j \leqslant K, \tag{6}$$

$$Y_{ij} \geqslant p_i + (X_{ij}-1)M \quad \forall i,j \text{ and } i \neq j, \tag{7}$$

$$Y_{ij} \leqslant p_i + (1-X_{ij})M \quad \forall i,j \text{ and } i \neq j, \tag{8}$$

$$Y_{ij} \leqslant MX_{ij} \quad \forall i,j \text{ and } i \neq j, \tag{9}$$

$$Y_{ij} \geqslant 0 \quad \forall i,j \text{ and } i \neq j, \tag{10}$$

$$\text{and} \quad (1), (3) \text{ and } (4).$$

In this model, constraint (6) is the constraint on F_2. Constraint sets (7)–(10) assure that if $X_{ij}=0$ then $Y_{ij}=0$ and if $X_{ij}=1$ then $Y_{ij}=p_i$, so that $Y_{ij}=p_iX_{ij}$ always holds. Unfortunately, the computational performance of this linearization is very poor because Lemma 1.4 no longer holds so we cannot relax the integrality constraint of X_{ij}. Therefore, we have to use computationally less efficient GAMS/BARON solver instead of GAMS/MINOS.

In this section, we defined the problem and proposed the SCP-based method that can utilize commercial NLP solvers to solve the SCP and generate an approximation for the efficient frontier for the problem. We also gave the LSCP model which can be solved to global optimum by the MINLP solvers. However, since the MINLP and NLP solvers are not yet widely used and they are not as much computationally efficient as LP solvers, we also aimed to develop an effective approximation method to find a set of efficient points. In the next section, we define a heuristic method to approximate the efficient frontier for the problem.

3. Cost index based approximation (CIBA) method

In Section 2, we showed some optimality properties and simplifying characteristics for the SCP. In this section, we further present another very important optimality property in Lemma 1.6. This property will be helpful to design a heuristic method. This property basically tells us that if a solution is locally optimal then we cannot improve the total manufacturing cost by only changing the processing times of the jobs. It can also be explained as follows: for a given job sequence there is a unique processing time vector $p^\star = (p_1^\star, p_2^\star, \ldots, p_N^\star)$ that minimizes the total manufacturing cost for a given total weighted completion time (K). This property is as follows:

Lemma 1.6. *For any job pair i, j, a local optimal solution must satisfy the following conditions*:

(i) *If* $p_i > p_i^{\text{l}}$ *and* $p_j > p_j^{\text{l}}$ *then* $\frac{\partial f_i(p_i)}{\partial p_i}\frac{1}{W_i} = \frac{\partial f_j(p_j)}{\partial p_j}\frac{1}{W_j}$, *where* $W_i = w_i + \sum_{k=1}^{N} X_{ik}w_k$ (*i.e. the sum of the weights of job i and jobs that job i precedes*).

(ii) *If* $p_i = p_i^{\text{l}}$ *and* $p_j > p_j^{\text{l}}$ *then* $\frac{\partial f_i(p_i)}{\partial p_i}\frac{1}{W_i} \geqslant \frac{\partial f_j(p_j)}{\partial p_j}\frac{1}{W_j}$.

Proof. Suppose that $\frac{\partial f_i(p_i)}{\partial p_i}\frac{1}{W_i} < \frac{\partial f_j(p_j)}{\partial p_j}\frac{1}{W_j}$ for some i, j. Then,

$$\lim_{\Delta p\to 0}\left(\frac{f_i(p_i+\Delta p)-f_i(p_i)}{W_i\Delta p}-\frac{f_j(p_j)-f_j\left(p_j-\frac{W_i}{W_j}\Delta p\right)}{W_j\frac{W_i}{W_j}\Delta p}\right)<0,$$

$$\lim_{\Delta p\to 0}\left(\frac{f_i(p_i+\Delta p)-f_i(p_i)+f_j(p_j)-f_j\left(p_j-\frac{W_i}{W_j}\Delta p\right)}{\Delta p}\right)<0.$$

Then, $\exists\Delta p>0$ s.t. $f_i(p_i+\Delta p)-f_i(p_i)+f_j(p_j)-f_j\left(p_j-\frac{W_i}{W_j}\Delta p\right)<0$, which means the current solution can be improved without violating the total weighted completion time constraint. This proves that if there exists a job pair that does not satisfy the given conditions, then the solution is not locally optimal. □

The heuristic approach (CIBA) starts with the solution Z_1 (Fig. 2) and generates new approximate efficient points by using the information in Lemma 1.6. After taking the solution Z_1, by slightly increasing the processing times of the jobs one at a time at each iteration, we decrease F_1 while increasing F_2. The critical issue is which job to choose to perturb at each iteration so that the achieved decrease in F_1 over the increase in F_2 is at the maximum (i.e. the 'biggest bang for the buck'). To make the most appropriate choice, we propose a new cost index r_i for each job i, such that

$$r_i=\frac{\partial f_i(p_i)}{\partial p_i}\frac{1}{W_i}.$$

It is the estimated manufacturing cost change per unit total weighted completion time loss to be achieved by increasing the processing time of job i. Note that this index definition comes from Lemma 1.6. Next, we choose the job with the minimum r_i value. It is important to note that $f_i(p_i)$ is decreasing and $r_i<0$ when $p_i^{\mathrm{l}}\leqslant p_i<p_i^{\mathrm{u}}$ for all i. Then, the processing time of the selected job is increased by a predefined Δ amount. Increasing processing time of a job may result in a schedule that violates Lemma 1.2. We check if the WSPT rule is violated or not and if so we reorder the jobs according to the WSPT rule. After reordering, F_2^{new} and F_1^{new} are calculated. For the updated and re-sequenced jobs, their r_i's are updated. This job selection and update process is applied until F_2^{new} reaches to K^{u}. Each schedule achieved at the end of an iteration is kept as an approximate efficient solution. By keeping Δ as small as possible we can achieve solutions which are more close to the optimality property defined by Lemma 1.6.

Having discussed the basic approach, the proposed cost index based approximation (CIBA) method is given below:

Step 1. Find the non-dominated solutions Z_1 and Z_2.
Step 2. Start with the solution Z_1, set $F_1^{\mathrm{new}}=F_1(Z_1)$ and $F_2^{\mathrm{new}}=F_2(Z_1)$. While $F_2^{\mathrm{new}}<K^{\mathrm{u}}$ do the following:
Step 2.1. Select the job m with the minimum r_m. If there are more than one such jobs, select the last one in the sequence.
Step 2.2. Set $p_m=p_m+\Delta$.
Step 2.3. If the WSPT rule is violated, re-sequence the jobs by the WSPT rule.
Step 2.4. Update r_i indices for job m and for all other jobs whose position in sequence is changed in Step 2.3.
Step 2.5. Update $F_1^{\mathrm{new}}=F_1^{\mathrm{new}}-[f_m(p_m)-f_m(p_m+\Delta)]$ and recalculate F_2^{new}.
Step 2.6. Report the current schedule with F_1^{new} and F_2^{new} as an approximate efficient solution.

In each iteration of the CIBA method, we want to improve total manufacturing cost by slightly losing from the total weighted completion time. Since we want to minimize both criteria at the same time, we always prefer to update the job with the maximum manufacturing cost gain per unit increase in the total weighted completion time (Step 2.1). This is due to Lemma 1.6 which states the optimality conditions on p_i's. By choosing the job with minimum r_i each time we try to keep the achieved solutions to be close as possible to hold the conditions given in Lemma 1.6. After increasing the processing time of the selected job, the sequence of jobs is updated (Step 2.4), if necessary, by the WSPT rule. This is due to Lemma 1.2 which states that in an optimal schedule the job sequence must satisfy the WSPT rule. Then, for the perturbed job and for all re-sequenced jobs, r_i values are updated. At each iteration, the schedule achieved is reported as an approximate efficient solution. An important property that holds for the solution set generated by the CIBA is the following:

Lemma 1.7. *Each iteration of the CIBA method generates a new approximate efficient solution.*

Proof. As discussed earlier, we have a nonlinear convex manufacturing cost function and the processing times can take any real value between $p_i^{\mathrm{l}} \leqslant p_i \leqslant p_i^{\mathrm{u}}$. At each iteration of the CIBA method, the processing time of a selected job is increased by a Δ amount, that means the total manufacturing cost is strictly decreasing in each iteration as shown in Fig. 1. Moreover, the total completion time strictly increases even if the job sequence changes. Therefore, in each iteration we generate a new approximate efficient solution that cannot dominate previously generated solutions. □

The solution quality of the CIBA method depends on the selected Δ value. Since we are making decisions based on $\frac{\partial f_i(p_i)}{\partial p_i}$'s, if we choose a small Δ value, we get a better approximation to an efficient solution. Furthermore, a smaller Δ leads to more solution points to be generated which implies a better approximation of the efficient frontier.

4. Total completion time problem

All the models, properties and algorithms that we have given above also apply for the total completion time problem, which is a special case where the weights of jobs are equal. Instead, we present a new model for the total completion time problem. In this section, we proved that this new model holds the same properties as the previous one. Moreover, we performed a set of trial runs and showed that the new model is computationally more efficient in terms of the CPU times in solving the total completion time problem. In this model, we introduce a new binary decision variable X_{ij} to control if job i is assigned to position j in the sequence. The new formulation for the SCP is as below:

$$\min\ F_1: \quad \sum_{i=1}^{N} f_i(p_i) \tag{11}$$

$$\text{subject to} \quad \sum_{i=1}^{N}\sum_{j=1}^{N}(N-j+1)X_{ij}p_i \leqslant K, \tag{12}$$

$$\sum_{j=1}^{N} X_{ij} = 1 \quad \forall i, \tag{13}$$

$$\sum_{i=1}^{N} X_{ij} = 1 \quad \forall j, \tag{14}$$

$$p_i^{\mathrm{l}} \leqslant p_i \leqslant p_i^{\mathrm{u}} \quad \forall i, \tag{15}$$

$$X_{ij} \in \{0,1\} \quad \forall i,j. \tag{16}$$

In the above model, the objective function corresponds to the total manufacturing cost. The constraint (12) gives the total completion time. Constraints (13) and (14) are the assignment constraints which guarantee that each job is assigned to a position in the schedule and each position has a job assigned. Constraints (15) and (16) are same as in the previous model.

We could have used a similar model with the assignment variables X_{ij} to solve the weighted total completion time problem. In that case, we will not be able to use the property stated in Lemma 1.4, and hence the relaxed SCP model cannot be solved by the NLP solvers. It turns out that the way we model the problem affects the solver to be used and the computational requirements. We now check the characteristics of this new formulation as we did for the previous model before.

Lemma 2.1. *When $K \leqslant K^{\mathrm{u}}$, the total completion time constraint (12) must be tight at the optimal solution. This implies that the optimal schedule must satisfy the shortest processing time first (SPT) rule.*

The proof for Lemma 2.1 is similar to the proof of Lemma 1.2, so we skip it due to the space limitations. We next consider the NLP relaxation of the new SCP formulation and look for the optimality conditions for the relaxed problem.

Lemma 2.2. *For the relaxed SCP, in a local optimal solution, if $p_{i_1} < p_{i_2}$ for any job pair (i_1, i_2), then job i_1 must be assigned to an earlier position than i_2. This implies that a local optimal solution to the relaxed SCP must have binary X_{ij}'s.*

Proof. Suppose that we have a solution S for the relaxed problem where we consider two jobs i_1 and i_2 with processing times $p_{i_1} < p_{i_2}$. Further consider two positions j_1 and j_2 in the schedule such that $j_1 < j_2$. Suppose that assignment variables in S are as follows:

$$X_{i_1 j_1} = \rho_1 \quad \text{and} \quad X_{i_2 j_1} = \rho_2,$$
$$X_{i_1 j_2} = \xi_1 \quad \text{and} \quad X_{i_2 j_2} = \xi_2, \qquad \text{where } \rho_1, \rho_2, \xi_1 \text{ and } \xi_2 \text{ are positive.}$$

Since we relaxed the integrality constraint, a solution to the relaxed problem may contain non-binary X_{ij} values. This means, the same job could be allocated to multiple positions in the schedule, which is infeasible for our original problem.

Total completion time value calculated for S is as below:

$$F_2(S) = \sum_{i=1}^{N}\sum_{j=1}^{N}(N-j+1)X_{ij}p_i = \Phi + (N-j_1+1)[\rho_1 p_{i_1} + \rho_2 p_{i_2}] + (N-j_2+1)[\xi_1 p_{i_1} + \xi_2 p_{i_2}]$$
$$= \Phi + [(N-j_1+1)\rho_1 + (N-j_2+1)\xi_1]p_{i_1} + [(N-j_1+1)\rho_2 + (N-j_2+1)\xi_2]p_{i_2},$$

where Φ is a constant value. Suppose that without changing the processing times, we change the assignment variables to get a new solution S'. Setting $\delta = \min(\xi_1, \rho_2)$, new values for the assignment variables are as follows:

$$X_{i_1 j_1} = \rho_1 + \delta, \quad X_{i_2 j_1} = \rho_2 - \delta,$$
$$X_{i_1 j_2} = \xi_1 - \delta \quad \text{and} \quad X_{i_2 j_2} = \xi_2 + \delta.$$

By this arrangement we reallocate these two jobs to positions j_1 and j_2 such that we increase job i_1's ratio in the preceding position j_1 without disturbing the assignment constraints.

Total completion time of the solution after this arrangement is

$$F_2(S') = \Phi + [(N-j_1+1)(\rho_1+\delta) + (N-j_2+1)(\xi_1-\delta)]p_{i_1}$$
$$+ [(N-j_1+1)(\rho_2-\delta) + (N-j_2+1)(\xi_2+\delta)]p_{i_2}.$$

Then, $F_2(S') - F_2(S) = \delta(j_2 - j_1)(p_{i_1} - p_{i_2}) < 0$, because $j_2 > j_1$ and $p_{i_1} < p_{i_2}$.

This proves that there is always an integer optimal solution for the relaxed problem. □

According to Lemma 2.2, a local optimal solution must have integer X_{ij}'s. Although for the case where $p_{i_1} = p_{i_2}$, we could have alternative non-integer local optimal solutions. As in the previous formulation for the weighted case we do not need a MINLP solver to solve the relaxed problem, and an NLP solver would be sufficient.

Lemma 2.3. *The feasible region for the relaxed SCP is not a convex set.*

Proof. Consider two jobs i_1 and i_2 in a schedule called A_1. They are assigned at positions k and $k+1$, respectively. Their processing times are $p_{i_1} = s_1$ and $p_{i_2} = s_2$ where $s_1 < s_2$.

$F_2(A_1) = Q + (N-k+1)s_1 + (N-k)s_2 = K$, where Q is a constant.

Consider another schedule A_2 which is identical to A_1 except that job i_1 is at position $k+1$ and i_2 is at position k with processing times $p_{i_1} = q_1$ and $p_{i_2} = q_2$ where $q_2 < q_1$. $F_2(A_2) = Q + (N-k+1)q_2 + (N-k)q_1 = K$.

Next, define a point A which is a convex combination of A_1 and A_2 as follows:

$A = \lambda A_1 + (1-\lambda)A_2$ where $0 < \lambda < 1$. At point A, $p_{i_1} = \lambda s_1 + (1-\lambda)q_1$ and $p_{i_2} = \lambda s_2 + (1-\lambda)q_2$. Also, $X_{i_1 k} = \lambda$, $X_{i_1(k+1)} = (1-\lambda)$, $X_{i_2 k} = (1-\lambda)$ and $X_{i_2(k+1)} = \lambda$.

$$\begin{aligned} F_2(A) &= Q + [(N-k+1)\lambda + (N-k)(1-\lambda)]p_{i_1} + [(N-k+1)(1-\lambda) + (N-k)\lambda]p_{i_2} \\ &= Q + \lambda^2[(N-k+1)s_1 + (N-k)s_2] + (1-\lambda)^2[(N-k+1)q_2 + (N-k)q_1] \\ &\quad + \lambda(1-\lambda)[(N-k+1)q_1 + (N-k)q_2] + \lambda(1-\lambda)[(N-k+1)s_2 + (N-k)s_1] > K \end{aligned}$$

since $s_1 < s_2$ and $q_2 < q_1$.

This example shows that the feasible region for the problem is not convex and this implies that a local optimum found by an NLP solver may not be the global optimum. The complexity discussion for the weighted case in Section 2 holds for this case, too. □

The SCP model for the total completion time case also includes nonlinear terms in constraint (12). We can linearize constraint (12) in the model by replacing the nonlinear term $p_i X_{ij}$ in constraint (12) with a variable Y_{ij} and adding constraints (7)–(10) to the model as we did in Section 2. By this way we can achieve the LSCP model for the unweighted case and solve it to global optimum by using the MINLP solver GAMS/BARON.

Next, we give another property analogous to Lemma 1.6.

Lemma 2.4. *For any job pair i, j a local optimal solution must satisfy the following conditions*:

(i) *If* $p_i > p_i^{\mathrm{l}}$ *and* $p_j > p_j^{\mathrm{l}}$ *then* $\frac{\partial f_i(p_i)}{\partial p_i}\frac{1}{n_i} = \frac{\partial f_j(p_j)}{\partial p_j}\frac{1}{n_j}$, *where* $n_i = \sum_{j=1}^{N} X_{ij}(N-j+1)$.

(ii) *If* $p_i = p_i^{\mathrm{l}}$ *and* $p_j > p_j^{\mathrm{l}}$ *then* $\frac{\partial f_i(p_i)}{\partial p_i}\frac{1}{n_i} \geqslant \frac{\partial f_j(p_j)}{\partial p_j}\frac{1}{n_j}$.

Proof. Suppose that in a solution $\frac{\partial f_i(p_i)}{\partial p_i}\frac{1}{n_i} < \frac{\partial f_j(p_j)}{\partial p_j}\frac{1}{n_j}$ for jobs i, j. Then

$$\lim_{\Delta p \to 0}\left(\frac{f_i(p_i+\Delta p) - f_i(p_i)}{n_i \Delta p} - \frac{f_j(p_j) - f_j\left(p_j - \frac{n_i}{n_j}\Delta p\right)}{n_j \frac{n_i}{n_j}\Delta p}\right) < 0,$$

$$\lim_{\Delta p \to 0}\left(\frac{f_i(p_i+\Delta p) - f_i(p_i) + f_j(p_j) - f_j\left(p_j - \frac{n_i}{n_j}\Delta p\right)}{n_i \Delta p}\right) < 0.$$

Then, $\exists \Delta p > 0$ s.t. $f_i(p_i+\Delta p) - f_i(p_i) + f_j(p_j) - f_j\left(p_j - \frac{n_i}{n_j}\Delta p\right) < 0$, which means the current solution can be improved by increasing p_i by Δp and decreasing p_j by $\frac{n_i}{n_j}\Delta$. This proves that if a solution does not satisfy the conditions above, then it is not locally optimal. □

As we have shown that similar properties as the weighted case apply to the alternative formulation for the total completion time case we can employ similar approaches to find an approximation for the efficient frontier. We can generate approximate efficient solution set by solving the new model in the SCP-based method as described in Section 2. We can also employ our CIBA method by just modifying the cost index r_i as $r_i = \frac{\partial f_i(p_i)}{\partial p_i}\frac{1}{n_i}$ where n_i is the number of jobs that job i precedes including itself.

5. Numerical example

In this section, we give a numerical example for the total weighted completion time problem and apply the SCP-based method and the CIBA method to generate an approximate efficient frontier. In this problem we have five jobs and the design attributes (D_i, L_i, d_i and S_i) of each job along with the required cutting tool type are given in Table 1. We first calculated p_i^{l} and p_i^{u} values for each job as given in the same table. They are used to define a manufacturing cost function for each job. For example, the manufacturing cost function for job 1 is $f_1 = 0.25 \times p_1 + 0.35 \times p_1^{-1.32}$

To find the solution Z_1 and the corresponding schedule, we first set all jobs' processing times to their lower bounds and apply the WSPT rule as shown in Table 2. Total manufacturing cost, F_1, for this initial schedule is 5.105 and the corresponding optimal total weighted completion time, F_2, is 4.820. The initial schedule gives us

Table 1
Specifications of the jobs in the numerical example

Job	D_i	L_i	d_i	S_i	Tool	w_i	p_i^l	p_i^u
1	1.9	4.6	0.211	168	5	1.2	0.295	1.302
2	2.0	4.9	0.151	156	1	1.3	0.447	1.138
3	1.6	4.3	0.204	180	9	1.1	0.297	0.594
4	1.9	4.6	0.138	156	5	1.9	0.203	1.029
5	1.6	4.2	0.170	175	9	1.0	0.251	0.530

Table 2
Schedules at Z_1 and Z_2

Position	Z_1					Z_2				
	Job	p	w/p	r_i	$f_i(p_i)$	Job	p	w/p	r_i	$f_i(p_i)$
1	4	0.203	9.38	−1.63	1.713	5	0.530	1.89	0.0	0.210
2	1	0.295	4.06	−1.64	1.822	3	0.594	1.85	0.0	0.235
3	5	0.251	3.99	−0.49	0.341	4	1.029	1.85	0.0	0.452
4	3	0.297	3.71	−0.58	0.359	2	1.138	1.14	0.0	0.483
5	2	0.447	2.91	−1.68	0.870	1	1.302	0.92	0.0	0.572

the ideal total weighted completion time and the nadir manufacturing cost. We also have the schedule (Z_2) corresponding to the minimum manufacturing cost in Table 2. For the minimum cost settings, $F_1 = 1.952$ and $F_2 = 15.646$. This schedule gives us the ideal manufacturing cost and the nadir total weighted completion time.

As stated in the proposed CIBA method, we start with the solution Z_1, and perturb the job with the minimum r_i in each iteration, such as job 2 at the first iteration. In this example, we set the step size $\Delta = 0.1$. In Table 3, we present the perturbed job (j), and after each perturbation the new p_j, w_j/p_j, sequence and r_i's along with the F_1 and F_2 values. As an example, after we perturb job 1 in iteration 2, we need to re-sequence the jobs to satisfy the WSPT rule. The algorithm progresses in this way by perturbing one job at a time until we reach to solution Z_2. As can be seen in Table 3, at each iteration we improve the total manufacturing cost (F_1) while losing from the total weighted completion time (F_2) (i.e. generate a new efficient solution) as discussed in Lemma 1.7.

Using the CIBA method, we generated 34 approximate efficient solutions for the example problem. As discussed before, we could also use a commercial NLP solver to find the minimum manufacturing cost for a given total weighted completion time level. In this study, we model the problem in GAMS and use the MINOS solver as the SCP-based method. In Table 4, there are two schedules generated for the total weighted completion time of 7.660. The first schedule is found by the CIBA method at iteration 10 given in Table 3. We also solved the same problem by MINOS. It can be seen from Table 4 that two schedules are different in terms of job sequences and job processing times. When we consider the schedule found by the SCP-based method using

Table 3
Results of the first 10 iterations by the CIBA method

Iter.	Pert. job (j)	p_j	w_j/p_j	Sequence	r_i					F_2	F_1
					1	2	3	4	5		
1	2	0.547	2.376	4 1 5 3 2	−1.64	−0.95	−0.58	−1.63	−0.49	4.950	4.939
2	1	0.395	3.037	4 5 3 1 2	−1.49	−0.95	−0.39	−1.63	−0.36	5.238	4.406
3	4	0.303	6.279	4 5 3 1 2	−1.49	−0.95	−0.39	−0.62	−0.36	5.888	3.748
4	1	0.495	2.423	4 5 3 1 2	−0.84	−0.95	−0.39	−0.62	−0.36	6.138	3.467
5	2	0.647	2.009	4 5 3 1 2	−0.84	−0.57	−0.39	−0.62	−0.36	6.268	3.370
6	1	0.595	2.016	4 5 3 1 2	−0.51	−0.57	−0.39	−0.62	−0.36	6.518	3.205
7	4	0.403	4.720	4 5 3 1 2	−0.51	−0.57	−0.39	−0.30	−0.36	7.168	2.923
8	2	0.747	1.740	4 5 3 1 2	−0.51	−0.34	−0.39	−0.30	−0.36	7.298	2.865
9	1	0.695	1.726	4 5 3 2 1	−0.68	−0.18	−0.39	−0.30	−0.36	7.540	2.761
10	1	0.795	1.509	4 5 3 2 1	−0.45	−0.18	−0.39	−0.30	−0.36	7.660	2.694

Table 4
Schedules generated by different methods when $F_2 = 7.660$

Position	Schedule by CIBA method					Schedule by SCP-based method				
	Job (i)	p_i	w_i/p_i	r_i	$f_i(p_i)$	Job (i)	p_i	w_i/p_i	r_i	$f_i(p_i)$
1	4	0.403	4.720	−0.30	0.773	4	0.395	4.81	−0.32	0.788
2	5	0.251	3.987	−0.36	0.341	5	0.261	3.83	−0.32	0.325
3	3	0.297	3.708	−0.39	0.359	3	0.316	3.48	−0.32	0.335
4	2	0.747	1.740	−0.18	0.549	1	0.704	1.70	−0.32	0.731
5	1	0.795	1.509	−0.45	0.672	2	0.763	1.70	−0.32	0.542

MINOS, we see that it satisfies Lemma 1.2, which states that total weighted completion time must be tight at optimality and it also satisfies the WSPT rule. The MINOS solution has integer X_{ij}'s, so that it gives a feasible schedule (Lemma 1.4). For this particular solution, the CIBA method gives a slightly better solution ($F_1 = 2.694$) than the GAMS/MINOS solver ($F_1 = 2.721$), which is an example for the case that a solution found by MINOS may not be a global optimal (Lemma 1.5). Finally, when we check the r_i values of the MINOS solution, we see that they satisfy Lemma 1.6 which states that a local optimal solution cannot be improved by just changing the processing times of jobs. In order to find the global optimum of F_1 for a given $F_2 = 7.660$, we solve the LSCP model by GAMS/BARON, which gives $F_1 = 2.664$ with the job sequence of 4–5–3–2–1 (the same sequence with the CIBA), and the processing times (0.402, 0.265, 0.321, 0.643, 0.886), respectively.

6. Computational results

In this study, we considered two bicriteria production optimization problems with a scheduling objective and manufacturing cost objective. The scheduling objectives we considered were total completion time and total weighted completion time. For each case, we provided two different efficient frontier approximation algorithms: namely the SCP-based method and the CIBA method. The SCP-based method is modeled in GAMS 2.5 using solver MINOS 5.51. The CIBA method is coded in C and compiled with Gnu C compiler. All codes were run on a computer with 1294 MB memory and Pentium III 1133 MHz CPU. In this section, we discuss the results of the computational study.

There are three experimental factors that can affect the efficiency of the proposed methods as listed in Table 5. The number of jobs is an important factor that affects the solution quality and computational requirements. The machine type is considered since different machines have different cutting power levels and different operating costs. Machines with higher maximum cutting power, H, are more expensive to buy so operating cost of them, C_o, is higher. C_o and H affect the p^l and p^u levels as well as the shape of the manufacturing cost function. C_t, the tooling cost level, affects the p^u and the shape of the manufacturing cost function too. We also consider 10 different cutting tool types with the specific coefficients given in Table 6. Each job can be manufactured by one of these cutting tools. For each experimental setting ($3 * 3 * 2$), we took five replications resulting in 90 different problem settings. Furthermore, we generated jobs' technical specifications as follows: D_i are selected from $U[1,4]$, L_i from $U[4,6]$, d_i from $U[0.05, 0.3]$, S_i from $U[150, 250]$, where $U[a,b]$ is a uniform distribution in interval $[a,b]$. For the weighted case we generated a weight for each job from $U[1,10]$. Finally, we used two different levels of step size Δ, such as 0.01 and 0.03.

We first present the results for the total weighted completion time problem. The number of approximate efficient points generated by the CIBA method depends on the experimental factors, job specifications and

Table 5
Experimental design factors

Factor	Definition	Level 1	Level 2	Level 3
N	Number of jobs	50	100	150
m/c	Machine type	$C_o = 1$, $H = 5$	$C_o = 2$, $H = 10$	$C_o = 4$, $H = 20$
C_t	Tooling cost level	$U[6,10]$	$U[15,19]$	

Table 6
Technical coefficients of the cutting tools

Tool	α	β	γ	K	b	c	e	C_m	g	h	l	C_s
1	4.0	1.40	1.16	40,960,000	0.91	0.78	0.75	2.394	−1.52	1.004	0.25	204,620,000
2	4.3	1.60	1.20	37,015,056	0.96	0.70	0.71	1.637	−1.60	1.005	0.30	259,500,000
3	3.7	1.30	1.10	13,767,340	0.90	0.75	0.72	2.315	−1.45	1.015	0.25	202,010,000
4	3.7	1.28	1.05	11,001,020	0.80	0.75	0.70	2.415	−1.63	1.052	0.30	205,740,000
5	4.1	1.26	1.05	48,724,925	0.80	0.77	0.69	2.545	−1.69	1.005	0.40	204,500,000
6	4.1	1.30	1.10	57,225,273	0.87	0.77	0.69	2.213	−1.55	1.005	0.25	202,220,000
7	3.7	1.30	1.05	13,767,340	0.83	0.75	0.73	2.321	−1.63	1.015	0.30	203,500,000
8	3.8	1.20	1.05	23,451,637	0.88	0.83	0.72	2.321	−1.55	1.016	0.18	213,570,000
9	4.2	1.65	1.20	56,158,018	0.90	0.78	0.65	1.706	−1.54	1.104	0.32	211,825,000
10	3.8	1.20	1.05	23,451,637	0.81	0.75	0.72	2.298	−1.55	1.016	0.18	203,500,000

Δ. We cannot determine the number of points to be generated by the CIBA method in advance. Therefore, to compare these two approaches we first run the CIBA method and generate a set of approximate efficient points. Then, we choose a subset of this solution set and run the SCP-based model for this subset. As discussed earlier, we generate points with total weighted completion time values in $[K^l, K^u]$. Out of these points, we chose 50 points other than Z_1 and Z_2 such that each successive point pair has equal (or almost equal) separation. This is because we want to test the algorithms at different total weighted completion time levels along the efficient frontier.

We measured the relative difference between the F_1 values for a given F_2 value, $R = (F_1(\text{CIBA}) - F_1(\text{SCP}))/F_1(\text{SCP})$. Another critical issue to consider is the computational requirements of both methods. In Table 7, we summarize the R level and the required CPUs. The given CPUs in this table are measured for the entire solution sets. The data shows that the relative difference between two methods on the average is very small in favor of the CIBA. We can say that on the average the CIBA performs slightly better than the commercial NLP solver MINOS in terms of the solution quality. The maximum R values show that there are cases where the SCP-based method performs better. Moreover, we conclude that the CIBA method can find significantly higher number of efficient points than the SCP-based method in a much shorter computational time. For $\Delta = 0.01$, the SCP-based method used 352.91 CPU seconds on the average to solve for 50 solution points, but the CIBA method just spent 29.56 CPU seconds on the average to generate 354,597 points. In an eighth of MINOS' CPU requirement, the CIBA method can generate 7000 times more alternatives. It is important to note that we originally had a nonlinear mixed-integer programming formulation. Due to Lemmas 1.4 and 2.2, we were able to solve these problems by using the MINOS solver by relaxing the integrality requirements. Still, the CPU needed to solve the SCP-model is quite high. As expected, when we increase the step size, Δ, we loose from the solution quality at each point but gain from the CPU time. Moreover, for a higher Δ value, the size of the approximate efficient solution set decreases. When we check the results for different levels of N, we observe that increase in the number of jobs increases the SCP-based method's CPU much more than the CIBA method, but the relative difference between two methods, R, slightly improves in favor of the CIBA method.

Table 7
Performance measures for the weighted case

Δ		Min	Max	Mean	Std. dev.
0.01	R	−0.003970	0.001559	−0.000108	0.000665
	SCP-based CPU	16.86	961.59	352.91	356.88
	CIBA CPU	0.78	125.59	29.56	35.17
	CIBA set size	32,084	1,066,094	354,597	296,026
0.03	R	−0.000936	0.000216	−0.000041	0.000177
	SCP-based CPU	17.30	955.57	354.19	355.70
	CIBA CPU	0.26	41.24	9.76	11.56
	CIBA set size	10,693	355,344	118,188	98,674

Next, we discuss the computational results for the total completion time case. In Table 8, we present the results for the unweighted case for different Δ values. We see that the overall performance of CIBA is very close to the SCP-based method but the SCP-based method performs slightly better than the CIBA in terms of solution quality. The minimum R values indicate that there are cases where the CIBA performs better. Results show that both methods require less computation time for the unweighted case compared to the weighted case as expected. Moreover, the CIBA method generated less number of efficient points for the unweighted case. This is because when the weights of completion times are selected from the interval $[1, 10]$ we achieve a larger $[K^{\mathrm{l}}, K^{\mathrm{u}}]$ interval so that the CIBA can generate more points for the weighted case if the same Δ value is used as in the total completion time case.

Another important question is the absolute performance of the SCP-based method from the global minimum due to Lemmas 1.5 and 2.3. We could solve the LCSP model only for 5 and 8 job problems within the reasonable CPU times by using the GAMS/BARON solver version 7.2.3. We applied the same experimental setting as above. We took five replications for each setting. For each replication, we applied the SCP-based approach for five efficient points that the CIBA generated. So, we solved a total of 300 MINLP problems by BARON and MINOS. Table 9 shows the relative difference values for the SCP-based methods using MINOS versus BARON and the CIBA versus the SCP-based method using BARON. Results show that both MINOS and CIBA find solutions very close to global optimal. There are some cases where MINOS finds the global optimal. The computational requirements of MINOS and CIBA are negligible for the considered number of jobs levels. We observe that when we increase the number of jobs from 5 to 8, the CPU time required by BARON increases by 300 times.

Up to this point we have compared the pointwise quality of individual solutions generated by different methods. However, since this is a multi-objective optimization problem, we also check the approximation quality of solution sets generated by the CIBA and SCP-based methods. In the literature, there are different metrics used to compare the approximation quality of solution sets generated by different methods. In this paper, we will use three of them. The first one is the area method proposed by Zitzler and Thiele (1998), which measures the size of the objective value space covered by a solution set. The second metric is the coverage difference of two sets, $\mathrm{CD}(A, B)$, by Zitzler (1999), such that $\mathrm{CD}(A, B) = \mathrm{Area}(A \cup B) - \mathrm{Area}(B)$. This measure shows the contribution of solution set A to the area covered by solution set B. These two metrics use the ideal and nadir values of the objective functions in order to normalize the objective values of solutions and calculate

Table 8
Performance measures for the unweighted case

Δ		Min	Max	Mean	Std. dev.
0.01	R	−0.000076	0.001232	0.000313	0.000309
	SCP-based CPU	12.70	509.43	201.60	194.72
	CIBA CPU	0.01	0.14	0.06	0.03
	CIBA set size	1324	9731	4471.7	2286.3
0.03	R	−0.019599	0.010221	0.002610	0.003642
	SCP-based CPU	11.42	522.29	202.65	195.51
	CIBA CPU	0.01	0.05	0.021	0.01
	CIBA set size	444	3247	1494.6	763.2

Table 9
Comparison with the global optimal solutions

N		Min	Max	Mean	Std. dev.
5	R(CIBA-BARON)	0.000014	0.005895	0.001007	0.001433
	R(MINOS-BARON)	0.0	0.006673	0.001258	0.002132
	BARON CPU	5.51	13.32	10.07	1.79
8	R(CIBA-BARON)	0.000010	0.004125	0.000105	0.000109
	R(MINOS-BARON)	0.0	0.004552	0.000092	0.001348
	BARON CPU	993.90	4627.83	3010.53	943.08

Table 10
Comparison of the approximation algorithms for $\Delta = 0.01$

Metric	Mean	Min	Max
Area(CIBA)	0.843	0.816	0.867
Area(SCP)	0.833	0.805	0.856
CD(CIBA, SCP)	0.011	0.010	0.012
CD(SCP, CIBA)	0.000002	0.0	0.000029
P(CIBA, SCP)	0.995	0.975	1.000

the corresponding areas. In our problem, ideal and nadir values are achieved at solutions Z_1 and Z_2. The last metric we use is the probability, $P(A, B)$, that an algorithm, A, gives a better solution than another algorithm, B. This metric is proposed by Hansen and Jaszkiewicz (1998). It is calculated as $P(A, B) = \int_{u\in[0,1]} C(A(u), B(u))\,du$, where

$$C(A(u), B(u)) = \begin{cases} 1, & f(A(u)) < f(B(u)), \\ 1/2, & f(A(u)) = f(B(u)), \\ 0, & f(A(u)) > f(B(u)) \end{cases}$$

and $f(A(u)) = \min_{x\in A}\{\max(uF'_1(x), (1-u)F'_2(x))\}$ where $F'_1(x) = \frac{F_1(x)-F_1(Z_2)}{F_1(Z_1)-F_1(Z_2)}$ which is a normalization of F_1. Here u is the weight of the normalized objective function F_1 for the decision maker. The method basically tries a number of u values between 0 and 1 and estimates the decision maker's probability to choose a solution generated by method A. The results in Table 10 show that on the average area covered by the CIBA algorithm is larger than the area covered by the SCP-based method. Although there is a small difference, paired t-test results show that CIBA is significantly better than SCP-based method in terms of the area measure. When we check the coverage difference results, we see that the contribution of CIBA to the area covered by the SCP-based method is much more than the opposite measure. Finally, we check the probability measure, which shows that for the 99.5% of the cases on the average, the decision maker would prefer to implement a solution achieved by the CIBA method.

Computational results show that the CIBA method has almost same pointwise cost quality with the SCP-based approach despite the much less computational time requirement. More than that, the CIBA method can generate significantly higher number of efficient solutions than the SCP-based methods in a short computation time. As a result, the approximation quality measures that we calculated for both methods show that CIBA achieves significantly better approximations of the efficient frontiers than the SCP-based method.

7. Conclusion

In this study, we dealt with controllable processing times where processing time decisions affect the manufacturing cost as well as the scheduling performance. We considered the bicriteria problem with the objectives of minimizing the manufacturing cost and the total weighted completion time. We proposed a very effective single criterion model to find efficient solutions for different levels of total weighted completion time. This model can be solved to integer local optimality by just using a commercial NLP solver. Additionally, we have derived important optimality properties for the model which led us to design a very quick and effective algorithm which generates an approximate efficient solution set. Although we focus on the CNC turning machines for practical purposes, our results are valid for any nonlinear convex processing cost function. This study is an important step to integrate the process planning and scheduling decisions since it copes with job sequencing and processing time decisions simultaneously. For the future research, we consider the minimizing manufacturing cost objective with different scheduling objectives and different machine environments.

Acknowledgements

The authors thank the anonymous referees for suggestions on improving the paper. This work was supported in part by the Scientific and Technical Research Council of Turkey under grant #2211.

Appendix A. Single machining operation problem (SMOP)

SMOP is the manufacturing cost minimization problem for the turning operation. Decision variables for the problem are the cutting speed (v_i) and the feed rate (f_i). The job to be machined has certain specifications like job diameter, length, depth of cut and maximum allowable surface roughness and a selected cutting tool. A cutting tool has certain technical coefficients.

There are three constraints in the problem. The first one is the tool life constraint which comes from the limitation that each job can use at most one tool. The second constraint is the machine power constraint that comes from the maximum applicable cutting power by the machine. The last one, the surface roughness constraint, satisfies the surface quality requirement for the operation. The geometric programming model for the problem to minimize manufacturing cost (i.e. the sum of the operating and the tooling costs) for job i is as follows:

$$
\begin{aligned}
\text{Minimize} \quad & \text{Cost} = C_o \cdot p_i + C_t \cdot U_i = C_1 v_i^{-1} f_i^{-1} + C_2 v_i^{(\alpha-1)} f_i^{(\beta-1)} \\
\text{Subject to} \quad & C'_t v_i^{\alpha-1} f_i^{\beta-1} \leqslant 1 \quad \text{(Tool life constraint)}, \\
& C'_m v_i^{b} f_i^{c} \leqslant 1 \quad \text{(Machine power constraint)}, \\
& C'_s v_i^{g} f_i^{h} \leqslant 1 \quad \text{(Surface roughness constraint)}, \\
& v_i, f_i > 0,
\end{aligned}
$$

where $C_1 = \frac{\pi D_i L_i C_o}{12}$, $C_2 = \frac{\pi D_i L_i d_i^{\gamma} C_{ti}}{12 C_i^{TL}}$, $C'_t = \frac{\pi D_i L_i d_i^{\gamma}}{12 C_i^{TL}}$, $C'_m = \frac{C_m d_i^{e}}{H}$ and $C'_s = \frac{C_s d_i^{l}}{S_i}$.

Theorem 1 (Akturk and Avci, 1996). *At least one of the surface roughness and machine power constraints is binding at optimality for SMOP.*

Theorem 2 (Kayan and Akturk, 2005). *The surface roughness constraint must be tight at the optimal solution.*

References

Akturk, M.S., Avci, S., 1996. Tool allocation and machining conditions optimization for CNC machines. European Journal of Operational Research 94, 335–348.

Brooke, A., Kendrick, D., Meeraus, A., Raman, R., 2004. GAMS: A User's Guide. GAMS Development Corporation.

Chen, Z.L., Lu, Q., Tang, G., 1997. Single machine scheduling with discretely controllable processing times. Operations Research Letters 21, 69–76.

Choi, J.C., Bricker, D.L., 1996. Effectiveness of a geometric programming algorithm for optimization of machining economics models. Computers and Operations Research 23, 957–961.

Daniel Ng, C.T., Cheng, C.T.E., Kovalyov, M.Y., 2003. Batch scheduling with controllable setup and processing times to minimize total completion time. Journal of the Operational Research Society 54, 499–506.

Gopalakrishnan, B., Al-Khayyal, F., 1991. Machine parameter selection for turning with constraints: An analytical approach based on geometric programming. International Journal of Production Research 29 (9), 1897–1908.

Hansen, M.P., Jaszkiewicz, A., 1998. Evaluating the quality of approximations to the non-dominated set. IMM Technical Report, Technical University of Denmark, IMM-REP-1998-7.

Hoogeveen, H., 2005. Multicriteria scheduling. European Journal of Operational Research 167, 592–623.

Janiak, A., Kovalyov, M.Y., Kubiak, W., Werner, F., 2005. Positive half-products and scheduling with controllable processing times. European Journal of Operational Research 165, 416–422.

Karabati, S., Kouvelis, P., 1997. Flow-line scheduling problems with controllable processing times. IIE Transactions 29, 1–14.

Kayan, R.K., Akturk, M.S., 2005. A new bounding mechanism for the CNC machine scheduling problems with controllable processing times. European Journal of Operational Research 167, 624–643.

Lamond, B.F., Sodhi, M.S., 1997. Using tool life models to minimize processing time on a flexible machine. IIE Transactions 29, 611–621.

Malakooti, B., Deviprasad, J., 1989. An interactive multiple criteria approach for parameter selection in metal cutting. Operations Research 37 (5), 805–818.

Murty, K.G., Kabadi, S.N., 1987. Some NP-complete problems in quadratic and nonlinear programming. Mathematical Programming 39, 117–129.

Nowicki, E., Zdrzalka, S., 1990. A survey of results for sequencing problems with controllable processing times. Discrete Applied Mathematics 26, 271–287.

Shabtay, D., Kaspi, M., 2004. Minimizing the total weighted flow time in a single machine with controllable processing times. Computers and Operations Research 31, 2279–2289.

Smith, W.E., 1956. Various optimizers for single stage production. Naval Research Logistics Quarterly 3, 59–66.
Tuyttens, D., Teghem, J., Fortemps, P.H., Van Nieuwenhuyze, K., 2000. Performance of the MOSA method for the bicriteria assignment problem. Journal of Heuristics 6, 295–310.
Vickson, R.G., 1980a. Choosing the job sequence and processing times to minimize processing plus flow cost on a single machine. Operations Research 28, 1155–1167.
Vickson, R.G., 1980b. Two single-machine sequencing problems involving controllable job processing times. AIEE Transactions 12, 258–262.
Wan, G., Yen, B.P.-C., Li, C.-L., 2001. Single machine scheduling to minimize total compression plus weighted flow cost is NP-hard. Information Processing Letters 79, 273–280.
Wu, J., Azarm, S., 2001. Metrics for quality assessment of a multi-objective design optimization solution set. Journal of Mechanical Design 123 (1), 18–25.
Zitzler, E., 1999. Evolutionary algorithms for multi-objective optimization: Methods and applications. Ph.D. dissertation, Shaker Verlag, Aachen, Germany.
Zitzler, E., Thiele, L., 1998. Multiobjective optimization using evolutionary algorithms—A comparative case study. In: Eiben, A.E. et al. (Eds.), Parallel Problem Solving from Nature (PPSN V). Springer, Berlin, Germany, pp. 292–301.
Zitzler, E., Thiele, L., Laumans, M., Fonseca, C., Fonseca, V.G., 2003. Performance assessment of multi-objective optimizers: An analysis and review. IEEE Transactions on Evolutionary Computation 7 (2), 117–132.