

# Common Due-Date Assignment and Job Scheduling on Single Machine and Parallel Machines

J. K. MANE<sup>1</sup> AND K. P. GHADLE<sup>2</sup>

<sup>1,2</sup>Department of Mathematics, Dr. Babasaheb Ambedkar Marathwada University, Aurangabad, Maharashtra, India-431004

Email: janardhan.mane1@gmail.com, drkp.ghadle@yahoo.com

**Keywords:** scheduling, sequencing, heuristic, due date assignment.

**Abstract:** In this paper we consider a total penalty for the  $n$  job, one machine scheduling problem in which all jobs have a common due date. This penalty function is based on the due date value and on the earliness or the lateness of each job in the selected sequence. The main objective is to determine the optimal value of this due date and an optimal sequence to minimize a total penalty function. We prove that the optimal due date result can be generalized to the parallel machine problem. The problem of simultaneously available jobs on several parallel and identical machines. The problem is to find the optimal due date, assuming this to be the same for all jobs and we present a simple heuristic to find an approximate solution. On the basis of a limited experiment, we observe that the heuristic is very effective solution.

## Introduction

We consider the basic one machine sequencing problem under the common assumptions listed in Baker [1]. All jobs have a common (but unknown) due date. The objective is to find an optimal value of the due date and an optimal sequence which minimizes the total penalty based on the due date value and the earliness or tardiness of each job. The few studies analyzing due date selection procedures that of Baker and Bertrand [2] are the most recent and directly related to the present one. They considered (among other things) a static one machine problem. One of the due date assignment procedures involved in their study was the common due date.

In this paper, we consider the basic problem of parallel-machine scheduling under stable conditions, which are characterized by the assumptions discussed in detail by various authors for example, Conway et al.[3] Rinnooy Kan[4] and French[5] Given a set of simultaneously available jobs, each assigned a common due-date, our objective is to find the optimal due-date and to sequence the jobs on the machines in an optimal fashion to minimize a penalty function. The penalty function depends on several relevant cost factors which are related to the values of the assigned due-date, as well as the individual job earliness and tardiness values.

The problem of optimal assignment of a common due-date and sequencing  $n$  jobs on a single machine has been considered by Panwalkar et al. [6] they show that the optimal common due-date value must coincide with one of the job completion times. They also present an efficient scheduling algorithm to solve the sequencing problem.

Most studies involving scheduling with due dates have treated due dates as exogenous decisions. Other job shop studies using various due dates have been reported in the literature (see for example, Eilon and Chaudhary [7], Jones [8], Weeks [9], and Weeks and Fryer [10]). Some of these studies include specified common due dates for the one machine problem with minimum total tardiness criterion.

In this paper (due date, earliness, and tardiness) are all linear. Linear cost functions present a case that is more tractable than that occurring with nonlinear costs. Two of the costs, due date and tardiness, are related to customer objectives while the earliness cost can be viewed as an important element that concerns the production shop and also present a simple proof of the optimality result for the single-machine problem, and then generalize the result to the parallel-machine problem. While in the single-machine case the job-sequencing problem can easily be

solved, this problem is too complex to be solved analytically under the parallel-machine situation. We therefore propose a simple heuristic to help determine a near-optimal solution to such a problem.

Their objective, however, was only to find the lowest value of due date such that no job would be tardy.

**Notations**

- N - Number of Jobs;
- M – Parallel machines;
- d\* - Optimal common due date;
- $\sigma^*$  - Optimal job sequence;
- P – Non-negative constants;
- $\sigma$  - Arbitrary member of  $\Pi$ ;
- $\Pi$  - All possible job sequences;
- $C_j[i]$  - Completion time of job  $j[i]$ ;
- $E_{j[i]}$  -Earliness of job  $j[i]$ ;
- $T_{j[i]}$  -Tardiness of job  $j[i]$ .

**Problem formulation**

Let  $N = \{1, 2 \dots n\}$  be a set of n independent jobs and M a set of m parallel and identical machines. Job  $i$  requires  $t_i$  amount of processing time on one of the machine,  $\forall i \in N$ . we assume that the job labeling in such that  $t_1 \leq t_2 \leq \dots \leq t_n$  . i.e., sequence 1, 2... ,n represents the SPT sequence. All jobs are assumed to have a common due date  $d$  .

Let  $\Pi$  be the set of all possible job sequences. Also let  $\sigma$  denote an arbitrary member of  $\Pi$  , and the subscript  $j[i]$  denote the job in position  $[i]$  on machine  $j$ .  $\forall [i] \in N, \forall j \in M$  , in sequence  $\sigma$  . It  $C_{j[i]}$  ,  $E_{j[i]}$  and  $T_{j[i]}$  denote the completion time, earliness and tardiness of job  $j[i]$  in  $\sigma$  ,then

$$E_{j[i]} = \max(0, d - C_{j[i]}) \text{ and } T_{j[i]} = \max(0, C_{j[i]} - d)$$

Our objective is to find the optimal common due-date,  $d^*$  and the optimal job sequence,  $\sigma^*$  such that some jobs are early and others are late so as to minimize a penalty function given by

$$f(d, \sigma) = \sum_{j \in M} \sum_{i \in N} \{P_1 d + P_2 E_{j[i]} + P_3 T_{j[i]}\} \tag{1}$$

$$= nP_1 d + P_2 \sum_{i \in N} (d - C_{[i]}) + P_3 \sum_{i \in N} (C_{[i]} - d) \tag{2}$$

Where  $P_1$  ,  $P_2$  and  $P_3$  are non-negative constants representing the due-date assignment cost per unit time, the earliness and tardiness costs per unit time, respectively.

**Preliminary Analysis.**

We now prove the following two lemmas related to the optimal due date value. Throughout the reminders of this section we assume  $P_1 < P_3$  .

**Lemma1.** For any specified sequence  $\sigma$ , there exists an optimal value of d which coincides with the completion time of one of the jobs in  $\sigma$  .

**Proof.** Suppose  $d < C_{[1]}$  . Then all jobs are tardy. If the due date is increased to  $C_{[1]}$  the total penalty will increase by  $n(P_1 \geq P_3)(C_{[1]} - d)$ . This quantity is non-positive.

Now suppose

$$C_{[i]} < d < C_{[i+1]} \quad \text{for } i = 1, 2, \dots, n-1.$$

Let  $x = d - C_{[i]}$  and Let  $y = C_{[i+1]} - d$  so that  $x > 0$  and  $y > 0$ .

If the due date is changed to  $C_{[i]}$ , the new penalty is given by

$$f(C_{[i]}, \sigma) = f(d, \sigma) + x(n(P_3 - P_1) - i(P_2 + P_3))$$

Similarly, if the due date is changed to  $C_{[i+1]}$ , the penalty is given by

$$f(C_{[i+1]}, \sigma) = f(d, \sigma) + y(n(P_3 - P_1) - i(P_2 + P_3))$$

Obviously,  $f(C_{[i]}, \sigma) \leq f(d, \sigma)$  if  $(n(P_3 - P_1) - i(P_2 + P_3)) \leq 0$  and  $f(C_{[i+1]}, \sigma) < f(d, \sigma)$  otherwise.

It is now clear that an optimal value of  $d$  coincides with the completion time of a job.

**Lemma 2.** For any specified sequence  $\sigma$ . There exists an optimal due date equal to  $C_{[k]}$ , where  $k$  is the smallest integral value greater than or equal to  $n(P_3 - P_1) / (P_2 + P_3)$ .

**Proof.** From lemma 1 we know that an optimal due date value can coincide with the completion time of some job. Let this be the job in position  $k$  where  $1 \leq k \leq n$ .

The total penalty is then equal to  $f(C_{[k]}, \sigma)$  since this due date is optimal, we know that for any  $\Delta > 0$

$$f(C_{[k]} + \Delta, \sigma) - f(C_{[k]}, \sigma) \geq 0$$

The right shift of the due date from  $C_{[k]}$  to  $C_{[k]} + \Delta$  causes an increase in the first two components of the total penalty by at least  $\Delta(nP_1 + KP_2)$  and causes a decrease in the third component by no more than  $\Delta(n - K)P_3$ .

Therefore,

$$\Delta(nP_1 + KP_2 - (n - K)P_3) \geq 0$$

Or

$$K \geq n(P_3 - P_1) / (P_2 + P_3) \dots \quad (3)$$

Also

$$f(C_{[k]} - \Delta, \sigma) - f(C_{[k]}, \sigma) \geq 0,$$

Or, the left shift of the due date results in

$$K - 1 \leq n(P_3 - P_1) / (P_2 + P_3)$$

Since  $K$  is an integer, the proof of lemma 2 follows immediately.

From the discussion thus far, it is clear that for any sequence, exactly  $k$  jobs will be non-tardy ( $K = 0$  if  $P_1 \geq P_3$ ).

The total penalty is equal to

$$f(C_{[k]}, \sigma) = \sum_{i=1}^n (P_1 d + P_2 E_{[i]} + P_3 T_{[i]}).$$

Substituting  $C_{[k]} = t_{[1]} + t_{[2]} + t_{[3]} + \dots + t_{[k]}$  and simplifying. We get,

$$f(C_{[k]}, \sigma) = \sum_{j=1}^k (nP_1 + (j-1)P_2)t_{[j]} + \sum_{j=k+1}^n P_3(n+1-j)t_{[j]}$$

$$= f(C_{[k]}, \sigma) = \sum_{j=1}^k \gamma_j t_{[j]},$$

where  $\gamma_j$  is equal to  $nP_1 + (j-1)P_2$ .

For  $j \leq k$  and is equal to  $P_3(n+1-j)$  for  $j > k$ .

To find the optimal sequence, we need to find the minimal penalty among all  $\sigma \in \pi$ . The term  $\gamma_j$  may be viewed as the positional penalty for sequence position  $j$ .

**Lemma3.** Quantity  $\left( \sum_{j=1}^n \gamma_j t_{[j]} \right)$  is minimized by matching the smallest value of  $\gamma$  with the largest value of  $t$ , the next larger value of  $\gamma$  with the next smaller value of  $t$ , and so on.

### Single-Machine

#### Optimal Solution Procedure

The single machine scheduling problem involves scheduling a set of tasks to a single resource. This is accomplished by determining a sequence that includes each task, and then assigning the tasks to the resource. Each task can be given a priority, ready time, processing time and due date. The value of the performance measures can be computed based on this information and the sequence of tasks.

The algorithm presented here consists of two phases. Phase 1 finds the number of nontrade job ( $k$ ). Phase 2 calculates positional penalties  $\gamma$ , the optimal sequence, and the optimal due-date ( $d^*$ ).

#### Phase 1

**Step (i)** Set  $K' \leftarrow n(P_3 - P_1) / (P_2 + P_3)$ .

**Step (ii)** Check if  $K' > 0$ . if YES: go to step (iii) If No: Set  $d^* \leftarrow 0$ . SPT sequence is optimal STOP.

**Step (iii)** Check  $K'$  is an integer If YES: set  $K \leftarrow K'$ .

**Proceed to phase 2.** If NO: set  $K$  equal to the smallest integer value  $> K'$ .

#### Phase 2

**Step 2.1** Label position  $j (1 \leq j \leq n)$  as

$$\gamma_j = \begin{cases} nP_1 + (j-1)P_2 & 1 \leq j \leq k \\ (n+1-j)P_3 & K+1 \leq j \leq n. \end{cases}$$

**Step 2.2** Rank the positional labels  $\gamma_j$  in descending order of magnitude such that the largest  $\gamma_j$  is ranked 1 and the smallest  $\gamma_j$  is ranked n. Break ties arbitrarily .

**Step 2.3** Obtain the optimal sequence such that job  $i$  is scheduled in position  $j$  corresponding to  $\gamma_j$  ranked in position  $i$  .

**Step 2.4** Set  $d^* \leftarrow (t[1] + t[2] + \dots + t[K])$  STOP.

**Numerical Example**

Given Ten jobs with  $t_1 = 2, t_2 = 5, t_3 = 8, t_4 = 11, t_5 = 14, t_6 = 16, t_7 = 19, t_8 = 22, t_9 = 23,$  and  $t_{10} = 25$ . The penalties are  $P_1 = 5, P_2 = 10$  and  $P_3 = 15$ .

$\Rightarrow$  From phase 1 the algorithms, we get  $K' = 4$ . the Ten positional labels and their ranks are as indicated below.

**Table 1.**

Position j	1	2	3	4	5	6	7	8	9	10
$\gamma_j$	50	60	70	80	90	75	60	45	30	15
Rank i	7	5	4	2	1	3	6	8	9	10

Optimal sequence: 7, 5,4,2,1,3,6,8,9,10.

$$d^* = t_7 + t_5 + t_4 + t_2 = 49$$

One can easily verify that the total penalty is 6,755 units.

**Parallel-Machine:**

Parallel machine scheduling involves scheduling a set of tasks on two or more machines that work in parallel with each other. The machines perform identical operations and may or may not operate at the same pace. The basic analysis of the unrestricted version has been extended to models involving parallel machines. Hall [11], Sundararaghavan [12] examine the minimization of total absolute deviation with  $m$  identical machines operating in parallel. The multimachine procedure assigns the  $m$  longest jobs to different machines. Thereafter, the jobs are treated in non-increasing order of processing times and assigned  $2m$  at a time among the machines. After all the jobs are assigned, Algorithm 1 is used to sequence the jobs on each machine. In addition, the four key properties apply to the optimal solution of the multimachine model in the form

- I. On each machine, there is no inserted idle time.
- II. On each machine, the optimal schedule is V-shaped.
- III. On each machine, one job completes at time  $d$  .
- IV. The number of jobs assigned to each of the  $m$  machines is either  $[n / m]$  or  $[n / m] + 1$  (where  $[x]$  denotes the integer portion of  $x$ ). Let this number be denoted  $q$  . Then, on each Machine, the  $b$  th job in sequence completes at time  $d$ , where  $b$  the smallest integer is greater than or equal to  $q / 2$ .

The procedure to find the optimal due-date value for the parallel-machine problem can be derived in a similar fashion. Instead of differentiating (2). We take the first derivative of (1). The general penalty functions with respect to  $d$  , and equate the result to zero to obtain the first-order necessary condition for optimality given by (3). Thus the optimal due-date  $d^*$  should be set so

that the numbers of non-tardy jobs are related to each other in accordance with equation (3). For a given jobs sequence  $\sigma$ , let the jobs in  $\sigma$  be indexed according to non-decreasing completion time i.e.  $C_1 \leq C_2 \dots \leq C_n$ .

If the optimal due-date is set as  $d^* = C_r$ , then the necessary optimality condition equation (3), requires  $r$  to satisfy the following condition.

$$nP_1 + (r-1)P_2 - (n-r)P_3 = 0$$

$$\text{Or } r = n(P_3 - P_1) / (P_2 + P_3) + P_2 / (P_2 + P_3) \tag{4}$$

Here we see that the optimal due-date should be set equal to the  $r$ th completion time, where  $r$  is the least integer value not less than  $n(P_3 - P_1) / (P_2 + P_3)$  for a given  $\sigma$ . It is clear, in this case, that  $d^* = C_r$  is explicitly dependent on  $\sigma$ , whereas, in the single-machine case  $d^* = C[r]$  is a function of the job position  $r$  that is independent of  $\sigma$ . This is where the similarity between the single-machine and parallel machine problems stops and the difficulty of finding the point optimal due-date and job sequence starts.

**Heuristic Solution:**

A heuristic procedure to solve the parallel-machine problem therefore, we propose a simple heuristic based on the optimal results desired for the single machine problem. The heuristic is best described as an algorithm, as follows Heuristic algorithm.

**Step 1.** Construct a list  $s$  of schedulable jobs in which the jobs are arranged in non-decreasing order of the processing times  $t_i$ . Initially,  $|s| = n$ .

**Step 2.** Set  $K = \lceil n/m \rceil$ , where  $\lceil x \rceil$  is the smallest integer greater than or equal to  $x$ .

**Step 3.** Set  $r = \lceil K(P_3 - P_1) / (P_2 + P_3) \rceil$ .

**Step 4.** Set  $\lambda_{j[i]} = KP_1 + (i-1)P_2$  for  $1 \leq i \leq r$  and  $1 \leq j \leq m$ ;  $\lambda_{j[i]} = (K+1-i)P_3$  for  $r+1 \leq i \leq k$  and  $1 \leq j \leq m$ .

**Step 5.** Construct a list  $A$  of Position labels in which the labels are arranged in non-increasing order of  $\lambda_{j[i]}$ . Initially,  $|A| = mk$ .

**Step 6.** Assign the first job in  $s$  to the position corresponding to the first label in  $A$ .

**Step 7.** Delete the first job and the first position label from  $s$  and  $A$  respectively. If  $s \neq \phi$ , go to Step 6.

**Step 8.** A job sequence  $\sigma$  is generated. Calculate the completion time of the jobs in  $\sigma$  and reindex the jobs in non-decreasing order  $C_i$  i.e.  $C_1 \leq C_2 \leq C_3 \leq \dots \leq C_n$ . Set  $d^* = C_r$ ,

Where  $r = \lceil n(P_3 - P_1) / (P_2 + P_3) \rceil$ .

**Step 9.** Evaluate  $f(d^*, \sigma) = \sum_{j \in M} \sum_{i \in N} \{P_1 d^* + P_2 E_{j[i]} + P_3 T_{j[i]}\}$ .

So the heuristic first constructs a list,  $S$ , of schedulable jobs in non-decreasing processing times (step 1), and then calculates  $K$ , the average number of jobs per machine (step 2). The algorithm then treats each machine individually as though they are  $m$  independent single-machine problems. Applying the optimal results of the single-machine problem, the heuristic finds the optimal position  $r$  (step 3), and defines a position label for each individual machine in accordance with expression (7) (step 4). The next three steps (steps 5, 6 and 7) are followed to assign jobs to machine positions in such a way as to match the longest job with the position label having the smallest value, the second longest job with the second smallest label, and so on until the set  $S$  is exhausted. This results in a job sequence,  $a$ , from which the job completion times are

identified and  $d^*$  determined using expression (8), the optimal due-date for the parallel-machine problem. Once both the due-date value and job sequence are determined, the corresponding penalty function value  $f(d^*, \sigma)$  can be evaluated (step 9), which is taken as an approximation of the true optimal solution. So the percentage error of the heuristic solution is  $\alpha = (\text{heuristic solution} - \text{optimal solution}) / \text{optimal solution} \times 100 \%$ .

**Table 2.** Result of 2 test runs

Sr.No	No. machines m	No.jobs n	$\alpha = (\text{heuristic solution} - \text{optimal solution}) / \text{optimal solution} \times 100 \%$
1	3	8	6.62
2	3	10	8.56

Minimum  $\alpha = 6.62\%$ , Maximum  $\alpha = 8.56\%$  and Average  $\alpha = 7.59\%$

### Optimal Solution

The computational requirement to obtain the true optimal solution by complete enumeration is prohibitive even for a small problem. It can easily be established that complete enumeration, in

the worst case, requires  $\sum_{1 \leq i \leq m} n_i^{n_1+n_2+\dots+n_m} (n_1!)(n_2!) \dots (n_m!)$  number of enumerations, where  $n_i$  denotes the number of jobs assigned to machine  $i$ ,  $1 \leq i \leq m$ , and the summation is over all non-negative integer values of  $n_1, n_2, \dots, n_m$  such that  $\sum_{1 \leq i \leq m} n_i = n$ .

Alternatively, the problem can be formulated as a mixed zero-one mathematical programme as follows:

$$\text{Min } z = \sum_{1 \leq i \leq n} \{P_1 d + P_2 E_i + P_3 T_i\},$$

Subject to

$$C_i - d = T_i - E_i \quad \text{for } 1 \leq i \leq n$$

$$C_i = \sum_{1 \leq j \leq m} \sum_{1 \leq k \leq n} c_{jk} x_{ijk} \quad \text{for } 1 \leq i \leq n$$

$$C_{jk} = C_{j,k-1} + \sum_{1 \leq i \leq n} t_i x_{ijk} \quad \text{for } 1 \leq i \leq m, 1 \leq k \leq n$$

$$0 \leq \sum_{1 \leq i \leq n} x_{ijk} \leq 1 \quad \text{for } 1 \leq i \leq m, 1 \leq k \leq n$$

$$\sum_{1 \leq i \leq m} \sum_{1 \leq k \leq n} x_{ijk} = 1 \quad \text{for } 1 \leq i \leq n$$

$$x_{ijk} = 0 \text{ or } 1 \quad \text{for } 1 \leq i \leq n, 1 \leq j \leq m, 1 \leq k \leq n$$

$$C_{j0} = 0 \quad \text{for } 1 \leq i \leq m,$$

Where  $x_{ijk} = 1(0)$  indicates job  $i$  is (not) assigned to position  $k$  of machine  $j$ ;  $C_i$  is the completion time of job  $i$ ;  $C_{jk}$  is the completion time of the job in position  $k$  of machine  $j$ .

## Conclusion

We consider a total penalty for the  $n$  job, one machine scheduling problem in which all jobs have a common due date. The main objective is to determine the optimal value of this due date and an optimal sequence to minimize a total penalty function. Hence the problem belongs to class P. It is interesting to note that the determination of  $K$ , the number of nontardy jobs, does not require the values of the processing times of the  $n$  jobs. We also show that the problem of scheduling an independent jobs on  $m$  parallel and identical machines. We further prove that the optimal due-date result can be generalized to the parallel-machines problem. We observe that the heuristic is very effective solution

## References

- [1] BAKER, K. R. 1974. *Introduction to Sequencing and Scheduling*. John Wiley & Sons, New York.
- [2] BAKER, K. R., AND J. W. M. BERTRAND. 1980. *A Comparison of Due-Date Selection Rules, Industrial Engr.* Report 1980/02, University of Technology, Eindhoven, The Netherlands.
- [3] R. W. CONWAY, W. L. MAXWELL and L. W. MILLER (1967) *Theory of Scheduling*. Addison-Wesley, Reading, Mass.
- [4] A. H. G. RiNNooy KAN (1976) *Machine Scheduling Problems: Classification, Complexity and Computations*. Martinus Nijhoff, The Hague.
- [5] S. FRENCH (1 982) *Sequencing and Scheduling Ellis Horwood*, Chichester.
- [6] S. S. PANWALKAR, M. L. SMITH and A. SEIDMANN (1982) *Common due-date assignment to minimize total penalty for the one machine scheduling problem*. Opns Res.30, 391-399.
- [7] EILON S. AND I. J. CHOWDHURY. 1976. *Due-Dates in Job Shop Scheduling*. Int. J.Prod. Res. 14, 223-237.
- [8] JONES, C. H. 1973. An Economic Evaluation of Job Shop Dispatching Rules. Mgmt. Sci. 20, 293-307.
- [9] WEEKS, J. K. 1979. *A Simulation Study of Predictable Due-Dates*. Mgmt. Sci. 25, 363-373.
- [10] WEEKS, J. K., AND J. S. FRYER. 1977. *A Methodology for Assigning Minimum Cost Due-Dates*. Mgmt. Sci. 23, 872-881.
- [11] HALL, N. 1986. *Single and Multi-Processor Models for Minimizing Completion Time Variance*. Navai. Res. Logist. Quart. 33, 49-54.
- [12] SUNDARARAGHAVAN, P., AND M. AHED. 1984. *Minimizing the Sum of Absolute Lateness in Single Machine and Multimachine Scheduling*. Navai. Res. Logist. Quart. 31,325-333.