International Journal of Industrial Engineering Computations 2 (2011) 491–498

Contents lists available at GrowingScience

International Journal of Industrial Engineering Computations

homepage: www.GrowingScience.com/ijiec

A mathematical model for weighted tardy jobs scheduling problem with a batched delivery system

Mohammad Mahdavi Mazdeh[a*], Amir Hamidinia[a*] and **Ayatollah Karamouzian[a]**

[a]*School of Industrial Engineering, Iran University of Science and Technology, Tehran, Iran.*

A R T I C L E I N F O

Article history:
Received 20 January 2011
Received in revised form
April, 14, 2011
Accepted 14 April 2011
Available online
16 April 2011

Keywords:
Scheduling
Single-machine
Tardy jobs
Batched delivery system
SA algorithm

A B S T R A C T

This study investigates minimizing the number of weighted tardy jobs on a single machine when jobs are delivered to either customers or next station in various size batches. In real world, this issue may happen within a supply chain in which delivering goods to customers entails costs. Under such circumstances, keeping completed jobs to deliver in batches may result in reducing delivery costs; nevertheless, it may add to the tardy jobs, which in turn leads to higher costs. In literature review, minimizing the number of weighted tardy jobs is known as NP-Hard problem, so the present issue aiming at minimizing the costs of delivering, in addition to the aforementioned objective function, remains an NP-Hard problem. In this study, the issue is assessed where the customers are numerous, and a mathematical model is presented. We also present a meta-heuristic method based on simulated annealing (SA) and the performance of the SA is examined versus exact solutions.

1. Introduction

Tardy jobs normally lead to customer dissatisfaction and enterprises always look for solutions to minimize them. During the past few decades, there have been significant attempts on minimizing the number of weighted tardy jobs on a single machine. However, there are a few studies available about the jobs, which are supposed to be delivered as batch to customers when completed. In this case, jobs could be delivered immediately to customers after completion, or remain in the system waiting for completion of other job/s which are ultimately delivered as single batch to the relevant customer. In this batching model, each batch with different sizes from one job to all remains jobs. Obviously, jobs remaining in the system increase tardy jobs and increase tardy costs. On the other hand, this may lead to decrease frequency of delivery to customers resulting significantly lower delivery costs. Optimizing these two values makes the problem more complex but the results are more realistic. This problem may occur in a supply chain when the target is to deliver completed jobs to different customers. In addition, this can take place within a manufactory where it is necessary to apply several processes on one job in order to produce the final product. The jobs with completed processes on a single machine must be delivered in the next process as single or as a batch.

* Corresponding author. Tel.: +982173225002
E-mail: mazdeh@iust.ac.ir (M. Mahdavi Mazdeh)

doi: 10.5267/j.ijiec.2011.04.003

In order to minimize tardy jobs, the jobs are processed on a single machine that can process at most one job at a time, and each job has certain processing time, due date and weight. A job is called on time when its completion time is before its due date otherwise it is called as a tardy job. The aim of the present research is to study minimizing total weighted tardy jobs as well as delivery costs for jobs sent as batches to the customers if all are ready at time zero.

Moore (1968) explained that this problem can be solved in O(nlogn) in its simplest form, i.e. when all jobs are ready to process at time zero and their weights are the same with no delivery cost (1 | | ΣwUj). Lenstra et al. (1977) investigated the problem of minimizing the number of tardy jobs. According to their research, the minimization problem when there is a consideration of release date for jobs, $1|r_i|\sum w_i U_i$, is an NP-Hard problem. They, also, showed that even when all the weights are equal or the form is $1|r_i|\sum U_i$, again the problem is NP-Hard. Karp (1972) proved that the problem is still NP-Hard when all delivery times equals d or $1|d_i = d|\sum w_i U_i$. Lawler (1994) illustrated that when there is a simultaneous Pi≤Pj and wj≤wi, the relationship between the jobs the problem can be solved using Moore's solution. Lawler, also, considered the problem with preemption assumption, $1|prmp, r_i|\sum U_i$, and solved it by a dynamic programming algorithm with complexity O(n4) (Lawler, 1990). Carlier (1984) proved that the problem of $1|p_i = p, r_i|\sum U_i$ (i.e. all processing times are equal to p) might be solved in O(n3logn).

Since the under-investigation problem is NP-Hard, we propose specific methods to solve it, such as branch and bound and Meta-heuristic Algorithms. Villarreal and Bulfin (1983) provided a branch and bound method for solving $1||\sum w_i U_i$ problems with up to 50 jobs. Tang (1990) extended their solution to handle problems including up to 85 jobs. Furthermore, Potts and Van Wassenhove (1988) proposed a branch and bound solution to solve similar scheduling problems with 1000 jobs and, finally, Hallah and Bulfin (2003) increased the number of jobs to 2500 in their solution.

Potts and Kovalyov (2000) provided a comprehensive review on the scheduling problems when the jobs are processed or delivered in batches. They specifically focused on dynamic programming approaches to this type of problem. Furthermore, Mason and Anderson (1991) investigated the weighted flow time minimization problems in batch delivery systems and proposed a branch and bound solution to solve the resulted problem. Hall and Potts (2003) provided dynamic programming solutions for a range of scheduling problems with batched delivery systems. Finally, Mahdavi et al. (2007, 2008) suggested branch and bound algorithms for weighted sum of flow times in a batched delivery system in two cases. The first case considers all jobs are available at the time zero and in the presence of ready time and they compared their solutions with Hall and Potts's (2003) propositions.

Hall and Potts (2003) presented a similar case to Mahdavi's problem. However, regarding the difficulty of the problem, they made some assumptions to make it solvable as follows:

-Their major assumption is that jobs within a certain batch should be on time, which means there is no tardy job on batches.

- There is no idle time or preemption between two successive jobs on a single machine.

- Jobs for each customer are sent according to earliest due date (EDD) rule. This rule is only applicable when time available for all jobs delivered to a certain customer is the same or zero.

In this paper, we consider cases in which the weights of all jobs are associated with the same customer are equal. This paper is organized as follows. In section two, the problem statement is explained and a mathematical model is developed. In section three accuracy of the model is investigated and computational experiments are presented. In section four, a SA algorithm is proposed and its performance is investigated. Finally, concluding remarks and future research directions are given in the last section.

2. Problem definitions

Consider a scheduling problem where there are *N* jobs available at the time zero. The Jobs belong to *F* customers each of which has n_j $(1 \le j \le F)$ orders (i.e. jobs). Processing time P_{ij}, completion time C_{ij}, due date d_{ij} and weight W_{ij} are considered for each job where *i* is job number and *j* refers to a customer. A Job is called on time when its completion time is before its due date (i.e. $C_{ij} < d_{ij}$) otherwise it is called as a tardy job. In the present problem, preemption is not allowed and there is no set up time. Each job can be submitted to the relevant customer immediately after completion or it can wait for the next job/s in order to be delivered as a batch. For a delayed job *i* from customer *j*, the tardiness variable U_{ij} is defined such that it is equal to 1 when the job is tardy and it is equal to 0, otherwise. In other words, if $C_{ij} \le d_{ij}$ then $U_{ij} = 0,$ else $U_{ij} = 1.$ Note that the completion time of each job is equal to the completion time of the last job added to the batch (in other words, it is equal to the completion time of the batch); because jobs are not delivered to the related customer unless the last one is completed. On the other hand, delivery cost for a batch is considered in the problem, which is different for each customer and can be shown by *D*. Thus, the goal is to minimize the total delivery costs for batches plus total delay costs for tardy jobs. Based on equation developed by Graham at el. (1979), it can be shown as $1 \mid\mid \Sigma wiUi + \Sigma DjYj$ where *w* is the weight of each job and its value is given by delay cost for the job, *D* is delivery cost for each batch, and *Y* is a binary variable which will be explained later.

2.1 Proposed mathematical model

Before developing the mathematical model, it is necessary to explain more about the problem and ideas used. First, it is assumed that there are *N* empty batches (i.e. the total number of jobs). These primary batches are called the virtual batches and their order is important. Next, the jobs are allocated to them and the process is controlled by the constraints mentioned in this section and the objective function is calculated for each stage. In addition, as the jobs are assigned to batches randomly; it embraces all scenarios first, and there may be some jobs found within a batch, which do not belong to the same customer. To avoid this, we consider some limitations, which would be explained later. Certainly, a virtual batch with no job does not really exist and will be removed. Therefore, the order of real batches is determined.

After allocating the jobs, it is necessary to investigate completion time of each job and determine whether they are tardy or not. As previously noted the completion time of each job is equal to the completion time of the batch, because each job must wait for the last job of that batch.

It must be mentioned in this model that jobs for a certain customer have the same weight (i.e. delay cost), and various customers have different weights.

Variables

j is the customer number and $j = 1, \cdots, F$

i is the job number of each customer when nj indicates the number of jobs belonging to customer j and $i = 1, \cdots, n_i$

W_{ij} delay cost of the i_{th} job of the j_{th} customer

D_j delivery cost of batches belonging to the j_{th} customer

x_{ijk} equals 1 if the i_{th} job of the j_{th} customer is in k_{th} batch and equals 0 otherwise

N total number of all ordered jobs ($N = \sum_{j=1}^{F} n_j$)

k the batch number with $k = 1, \cdots, N$

y_{jk} equals 1 if there is a job belonging to the k_{th} batch which relates to the j_{th} customer and is equal to 0 otherwise

U_{ij} equals 1 if the i_{th} job belonging to the j_{th} customer is a delayed job and is equal to 0 otherwise

D_{ij} delivery time of the i_{th} job of the j_{th} customer

C_k completion time of the kth batch

C_{ij} completion time of the i_{th} job of the j_{th} customer

M A big number

Mathematical model :

$$\max \sum_i \sum_j W_{ij} U_{ij} + \sum_j \sum_k D_j y_{ij} \tag{1}$$

subject to

$$\sum_k x_{ijk} = 1 \qquad \forall i \in n_j, \forall j \in F \tag{2}$$

$$\begin{cases} C_k = \sum_{i=1}^{n_j} \sum_{j=1}^{F} x_{ijk} P_{ij} + C_{k-1} & \forall k \in (1, \cdots, n) \\ C_0 = 0 \end{cases} \tag{3}$$

$$C_{ij} = \sum_{k=1}^{F} C_k x_{ijk} \qquad \forall i \in n_j, j \in f \tag{4}$$

$$\begin{cases} -(C_{ij} - d_{ij}) + MU_{ij} \geq 0 & \forall i \in n_j, j \in f \\ (C_{ij} - d_{ij}) + M(1 - U_{ij}) \geq 0 & \forall i \in n_j, j \in f \end{cases} \tag{5}$$

$$\begin{cases} -\sum_i x_{ijk} + My_{jk} \geq 0 & \forall j \in f, \forall k \in (1, \cdots, N) \\ \sum_i x_{ijk} + M(1 - y_{jk}) \geq 0 \\ \sum_j y_{jk} \leq 1 & \forall k \in (1, \cdots, N) \end{cases} \tag{6}$$

$$x_{ijk}, U_{ij}, y_{jk} \in (0,1) \quad \forall i \in n_j, \forall j \in f, \forall k \in (1, \cdots, N) \tag{7}$$

The objective function given in Eq. (1) consists of two terms; the first term indicates the total delay costs based on the number of tardy jobs and the second term indicates total delivery costs. Eq. (2) shows that each job is placed only within one of the specified batches. If the 2nd job of the 1st customer is placed in the 3rd batch (i.e. $x_{213} = 1$), then all x_{21k} for $k \neq 3$ would be equal to zero. Eq. (3) is the recursive relationship, which determines completion time of each batch. It is equal to total processing time of jobs within that batch plus the completion time of previous batch. As each batch has a specific number, the order of all is remarkable. Eq. (4) indicates that since every job should wait for completing its batch, the completion time of each job is equal to the completion time of the batch. Eq. (5) specifies that the job is tardy if its completion time is after the delivery time (i.e. $C_{ij} - d_{ij} > 0$). This defines binary variable U_{ij} as 1 when job is tardy, otherwise as 0. Eq. (7) specifies that the number of batches is equivalent to the number of total jobs and it is possible to remove one or more supposed batches due to placing several jobs within a certain batch. The binary variable of y_{jk} is

defined such that it is 1 when a job from customer *j* is placed in the batch, and it is 0 when no job is placed. This constraint, furthermore, indicates that each batch is only and only belongs to a single customer, if exists. Finally, Eq. (7) indicates the binary values of x_{ijk}, U_{ij} and y_{jk}.

3. Experiments

In this section, we examine the performance of the proposed model of this paper using one numerical example solved by Lingo software.

3.1. Example

As Table 1 shows, the example includes two customers and seven jobs where *D* indicates the delivery cost and *W* is the delay cost for each tardy job.

Table 1

Specifications of Evaluation Case

Customer No.	1				2		
Job No.	1	2	3	4	1	2	3
Processing Time	6	10	12	17	10	16	18
Due Date	8	24	44	85	90	60	31

Two specific states are considered to test the model:

A) First, it is assumed that for both customers, delivery costs are much more than delay costs (i.e. D >> W). In this state, the delivery costs for each batch of customers 1 and 2 are equal to 110 and 120 and the delay costs are supposed as 15 and 20, respectively. Obviously, we expect that all jobs associated with one customer would be placed within a single batch. In this state, the optimal solution, given the small size of the problem, is easily obtained and compared with the result of Lingo. Table 2 presents the result of this calculation.

B) Now consider a case where the delay costs are much higher than the delivery costs (i.e. D<< W). In this case, the delay costs for customers 1 and 2 are equal to 120 and 130, and the delivery costs are supposed to be 20 and 18, respectively. Obviously, unlike the previous state, it is expected to be more batches. Table 3 presents the results of the solutions achieved by Lingo and its comparison by optimal solution. As we can observe, the results of the model are consistent with the optimal solutions through the two mentioned states. Hence, this confirms the accuracy of the mathematical model behavior.

Table 2

Results of Lingo

Input D_i	Input w_i	Global Optimum (obtained logically)	Result of Lingo	Running Time
$D_1 = 110$, $D_2 = 120$	$w_1 = 15$, $w_2 = 20$	310	310	195 sec
$D_1 = 20$, $D_2 = 18$	$w_1 = 120$, $w_2 = 130$	234	234	306 sec

3.2 Benefit of the batched delivery system

To show the benefit of the model from the economical point of view, two states are considered: 1) the state in which all jobs are delivered to the customers as soon as they are completed (i.e non-batching mode). In this state, Moore (1968) developed an algorithm to obtain the optimal solution when the weights (i.e delay costs) of all jobs are supposed to be the same, and 2) the state in which the batched delivery system is applied, using the proposed model of this paper.

Table 3
Results

Number of customers	Number of jobs	Inputs	Optimum cost for batching case (suggested model)	CPU time (lingo)	Optimum cost for non-batching case (Moor algorithm)
2	7	$D_1 = 80, D_2 = 90, w = 70$	520	200 sec	730
3	10	$D_1 = 70, D_2 = 80, D_3 = 60, w = 55$	540	580 sec	755
3	20	$D_1 = 65, D_2 = 80, D_3 = 60, w = 70$	1055	720 sec	1485

To compare these two states, we assume that all jobs have equal delay costs. First, we categorize the problems in terms of their sizes into two groups of small, and large. Examples with small sizes have maximum 10 jobs and 2 customers, while for the case of large-scale problems, there are more than 10 jobs and 2 customers. Processing times and delivery times for jobs are randomly generated between 1 to 300. In addition, delivery and delay costs for each customer are randomly generated between 1 to 100. As it is clear from Table 3, the Lingo software can solve the examples with small and medium sized with help of the mathematical model presented in less than 30 minutes. The results of three different examples are presented in this table, which compares batching and non-batching cases.

If the number of jobs or customers goes up, according to the complexity of the problem, then the running time will increase and practically the problem becomes difficult to solve. We can conclude that if the size of the problems in terms of the number of jobs or customers increases, the running time in Lingo software will exponentially increase, more resulting from non-linear relationship of the mathematical model. This leads to increase the number of nonlinear variables and increase the running time. Therefore, using heuristic and meta-heuristic methods such as simulated annealing (SA) algorithm are inevitable.

4. The proposed SA method

SA is a probabilistic method proposed by Kirkpatrick et al. (1983) and Cerny (1985) to find near-optimal solutions for some NP-Hard problems. It works by emulating the physical process whereby a solid is slowly cooled so that when eventually its structure is so called “frozen”, a minimum energy configuration happens. In the SA algorithm, solutions are encoded by a matrix depicted in Fig. 1 where the rows represent the batches and the columns represent the customers. For instance, if the element in row 2 and column 1 is one, the first order of customer 1 is assigned to batch 2. Note that the sum of elements in each column of this matrix is equal to 1. The main features of an SA algorithm are as follows:

- Generation of Initial solution,
- Generation of neighborhoods,
- Evaluation of new solutions,
- Acceptance of solutions according to a probability function,
- Equilibrium condition,
- Cooling schedule (Updating temperature),
- Stopping criteria,

The initial solution is generated by assigning each batch to an order. For generating a neighborhood, a column is selected, which shows an order of a customer. Then, the element with value one in this column is transferred to a new batch containing either no job or the other jobs of that customer. This structure of neighborhood considers the restriction of different jobs in batches (A batch cannot have

jobs belonging to different customers). The objective function of the model is applied for evaluation of fitness of solutions. The rest of features and parameters of the SA algorithm are listed in Table 4.

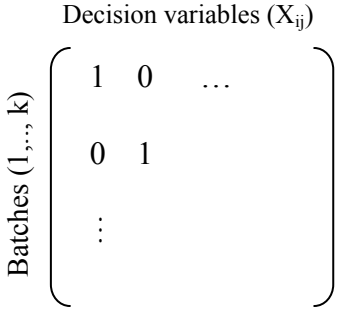

Fig.1. Solution representation

Table 4
The SA parameters/features

Parameters/ features	Value/function
Initial temperature	100
Final temperature	0
Epoc number	3
Cooling rate (α)	.02
Cooling schedule	$T_n = T_{n-1} - \alpha$
Acceptance probability function	$e^{-\Delta E/T}$

To evaluate the performance of the algorithm, the results of several instances with different sizes solved by two approaches, Lingo software and SA algorithm, are presented in Table 5. As it can be seen, the comparison of results shows that the SA algorithm solution provides an appropriate tool for scheduling problems with batch delivery systems.

Table 5
Comparison of lingo and SA outputs

Number of Problem	Problem size (Customer*order)	SA Objective Value				Solver (Lingo)		Gap
		Avg	Best	Std	Avg CPU Time (s)	Objective	Time (s)	
1	2*2	32	31	1	1.2	31	1	0.0%
2	2*4	122	119	7	4.0	119	173	0.0%
3	3*6	605	546	41	6.7	546	21600	0.0%
4	6*7	564	492	45	14.3	-	-	-

For each problem, the SA is run five times and the average and the best objective function, the average computational time, and the standard deviation of results are presented. To compare the results of the SA algorithm with optimal solutions obtained by solver, a quality criterion is defined according to the following equation:

$$Gap = \frac{(LINGO_{answer} - SA_{best\,solution})}{LINGO_{answer}} \times 100$$

5. Conclusion

We have presented a model to minimize the number of weighted tardy jobs on a single machine when jobs are delivered to the customers or next station as batched. In literature review, minimizing the number of weighted tardy jobs is known as NP-Hard problem, so the current issue that seeks to simultaneously minimize delivery costs besides of the objective function is still an NP-Hard problem.

The presented mathematical model faces nonlinear constraints; consequently, the optimal solution is not available using direct implementation of software packages such as Lingo. Therefore, we implemented a simulated annealing as a meta-heuristic to solve the resulted problem and the implementation of SA was compared with the solution of optimal solutions. Furthermore, regarding the low computation time required by the proposed SA algorithm, it seems suitable for real-time applications with similar properties.

References

Carlier, J. (1981). Probleme a une machine et algorithmes polynomiaux, Questio 5 (4).

Carlier, J. (1984). Problemes d'ordonnancements a contraintesde ressources: Algorithmes et complexite, These d'EEtat, Ph.D. Thesis, University of Paris VI.

Cerny, V. (1985). A thermodynamic approach to the traveling salesman problem: An efficient simulation. Journal of Optimization: Theory and Applications. 45**,** 41-51.

Graham, R. L., Lawler, E. L., Lenstra, J. K., & Rinnooy Kan, A. H. G. (1979). Optimization and approximation in deterministic machine scheduling: A survey, *Annals of Discrete Mathematics*, 5, 287–326.

Hall, N. G., & Potts, C. N. (2003). Supply chain scheduling: Batching and delivery. *Operations Research*, 51 (4) 566–584.

Hallah, R. M., & Bulfin, R. L. (2003). Minimizing the weighted number of tardy jobs on a single machine. *European Journal of Operational Research*, 145, 45–56.

Karp, R. M. (1972). *Reducibility among combinatorial problems, in: Miller, R.E., Thatcher, J.W. (Eds.), Complexity of Computer Computations*. Plenum Press, New York, 85–103.

Kirkpatrick, S., Gelett, C. D. and Vecchi, M. P. (1983). Optimization by simulated annealing. *Science,* 220*,* 621-630**.**

Lenstra, J. K., Rinnooy Kan, A.H.G., & Brucker, P. (1977). Complexity of machine scheduling problems, *Annals of Discrete Mathematics*, 1, 343–362.

Lawler, E. L. (1994). Knapsack-like scheduling problems, the Moore–Hodgson algorithm and the tower of sets property. *Mathematical Computer Modeling*, 20, 91–106.

Lawler, E. L. (1990). A dynamic programming algorithm for the preemptive scheduling of a single machine to minimize the number of late jobs. *Annals of Operations Research*, 26, 125–133.

Mason, A. J. & Anderson, E. J. (1991). Minimizing flow time on a single-machine with job classes and setup times. *Naval Research Logistics,* 38, 333–350.

Mahdavi-Mazdeh, M., Sarhadi, M., & Hindi, K. S. (2007). A branch-and-bound algorithm for single-machine scheduling with batch delivery minimizing flow times and delivery costs. *European Journal of Operational Research*, 183, 74–86.

Mahdavi-Mazdeha, M., Sarhadia, M., Hindi, K. S. (2008). A branch-and-bound algorithm for single-machine scheduling with batch delivery and job release times. *Computers & Operations Research*, 35, 1099 – 1111.

Moore, J. M. (1968). An n job one machine algorithm for minimizing the number of late jobs. *Management Science*, 15, 102–109.

Potts, C. N. & Kovalyov, M. Y. (2000). Scheduling with batching: A review, *European Journal of Operational Research*, 120, 228–249.

Potts, C. N., & Van Wassenhove, L.N. (1988). Algorithms for scheduling a single machine to minimize the weighted number of late jobs, *Management Science*, 34, 843–858.

Tang, G. (1990). A new branch and bound algorithm for minimizing the weighted number of tardy jobs, Annals *of Operations Research*, 24, 225–232.

Villarreal, F. J., & Bulfin, R. L. (1983). Scheduling a single machine to minimize the weighted number of tardy jobs. *IIE Transactions*, 15, 337–343.