Journal of Optimization in Industrial Engineering
Vol.15, Issue 2, Summer & Autumn 2022, 313- 321
DOI: 10.22094/JOIE.2022.1949101.1926

Research Article

Two-Machine Open Shop Scheduling with Proportionally Deteriorating Jobs and Makespan Objective

Ching-Fang Liaw*

Department of Industrial Engineering and Management, Chaoyang University of Technology, Taichung, Taiwan.

Received 07 January 2022; Revised 16 August 2022; Accepted 17 August 2022

Abstract

This manuscript examines the two-machine open shop scheduling problem where the latter a job is scheduled the longer it takes to process this job. The performance is measured by minimizing the makespan. By modifying existing algorithms for the corresponding problem with fixed processing times, two new algorithms are developed for the problem under consideration. The proofs of optimality of both algorithms are presented. The execution of these algorithms is illustrated by two numerical examples. Finally, both algorithms are further modified to solve a more generalized problem where the time demanded to process a job is a general linear function of its beginning time.

Keywords: Scheduling; Open shop; Deteriorating jobs; Makespan

1. Introduction

A traditional scheduling problem assumes fixed job processing times (Enayati et al. 2021, Yavari et al. 2020, Behnamian, 2019). Nevertheless, in many production occasions the time needed to process a job can vary and rely on the beginning time of its processing. In some cases, jobs may get deteriorated while waiting to be processed, which will increase their processing times. For instance, the time it needs to process an ingot in a steel mill will become longer as its waiting time increases. Also, at the scene of a fire, the delayed arrival of a fire engine will increase the time demanded to extinguish the fire. Other real life examples include the time required for maintaining a machine tends to grow proportionately to the time elapsed from the last maintenance, the time needed to remove the dust from an equipment will increase over time, the time demanded for the operation of a patient usually increases with the delay of the start of the operation, the travel time between two points in a congested urban area will gradually increase as it approaches the rush hour etc.

Scheduling problems for which the latter a job is scheduled the longer it needs to process this job are generally known as scheduling with deteriorating jobs, and have been broadly examined in the literature. Various models have been proposed for scheduling with deteriorating jobs including the sequence-related model and the time-dependent model. In a sequence-related model (Gawiejnowicz, 1996a; Strusevich and Rustogi 2017), the processing time of a job is determined by its order in the schedule. In other words, the time it needs to process a job relies on the number of its preceding jobs. The time-related model (Alidaee and Womer, 1999; Cheng et al., 2004; Gawiejnowicz, 1996b, 2008, 2020b) however presumes that the time it takes to process a job is a function of its beginning time. The problem examined in this manuscript belongs to the time-related model. Time-related models can be further classified into several different models depending on the form of the function which defines job processing times. In this manuscript, we address the proportional deterioration model where the job processing times proportionally deteriorate in time.

Current research concerning scheduling problems with deteriorating jobs mostly examines single-machine and parallel-machine problems. Even though a few researches have explored this issue for dedicated-machine problems, they have focused on flow shop models. The study of scheduling with deteriorating jobs for job shop or open shop problems has been very limited in literature. This manuscript examines the scheduling problem with deteriorating jobs for two-machine open shops.

Open shop is used to model the manufacturing environment where each job must visit each machine and there is no restriction on the routing of each job through the machines. In an open shop, a machine cannot simultaneously process exceeding one job and a job cannot be simultaneously processed by exceeding one machine. Examples of an open shop scheduling model include the scheduling of product testing activities and equipment maintenance operations, as the order in which the testing activities and the maintenance operations are performed is immaterial. Also, the open shop scheduling problem is often employed to model various problems occurring in the healthcare division. For example, patients need to go through a series of examinations and diagnoses, and the order in which these examinations and diagnoses are performed does not matter. The class-

*Corresponding author Email address: cfliaw@cyut.edu.tw

teacher scheduling problem is another case for which the open shop model is a commonly used formulation. The reader is referred to Almadian et al. (2021) for more applications of open shop models across different sectors.

The problem investigated in the manuscript is described as follows. A set of n independent jobs $N=\{1, 2,\ldots, n\}$ is required to be processed by two machines M_1 and M_2. Each job j can first visit machine M_1 and then machine M_2 or vice versa. We assume that preemption is not permitted and both machines are constantly available from time $t_0>0$ for processing jobs. The time demanded to process job j on machine M_i is $p_{ij}=b_{ij}t$, where $b_{ij}>0$ denotes the deterioration rate of job j when it is processed by machine M_i and $t> t_0$ denotes the beginning time of the processing of job j on machine M_i, for $i=1, 2$ and $j=1, 2,\ldots, n$. The performance measure to be minimized is makespan, C_{max}. Using the three-column symbolization of scheduling problem (Graham et al., 1979), the two-machine open shop scheduling problem with proportionally deteriorating jobs is denoted as $O_2/p_{ij}=b_{ij}t/C_{max}$.

The remainder of this manuscript is organized as follows. A review of preceding research focusing on the scheduling of two-machine open shops with fixed and variable processing times to minimize makespan is given in the next section. By modifying existing algorithms for the problem where processing times are fixed, two new algorithms for the problem where processing times proportionally deteriorate in time, $O_2/p_{ij}=b_{ij}t/C_{max}$, are presented and analysed in section 3. Then, section 4 examines the extension of the proposed algorithms for solving a more generalized problem where the time it takes to process a job is proportional to a linear function of its beginning time. Finally, conclusions along with some possible future research directions are presented in section 5.

2. Literature Review

The open shop scheduling problem where processing times are fixed and the performance measure to be minimized is makespan was first examined by Gonzalez and Sahni (1976). They showed that an optimal schedule for the two-machine problem $O_2//C_{max}$ can be found in $O(n)$ time. The problem $O_m//C_{max}$ with $m\geq 3$ is NP-hard. The problem $O_2//C_{max}$ has a special structure that allows different solution methods. Gonzalez and Sahni first proposed an $O(n)$-time algorithm (hereafter called algorithm *GS*) for this problem. Algorithm *GS* starts with partitioning the set of jobs into two subsets. Then, a particular job, called the diagonal job, is selected. Finally, an optimal schedule is constructed by combining a flow shop type schedule with the processing of the diagonal job. Pinedo and Schrage (1982) developed the Longest Alternate Processing Time (*LAPT*) rule and verified its optimality for the problem $O_2//C_{max}$. The key to the *LAPT* rule is the priority of a job processed by one machine is decided by its remaining processing time on the other machine. de Werra (1989) presented a two-phase algorithm (hereafter called algorithm *W*) for the same problem. Algorithm *W* splits the set of n jobs into three batches and finds an optimal schedule to the sequential batch processing problem (Gribkovskaia et al., 2006). Finally, Soper (2015) introduced another two-phase algorithm (hereafter called algorithm *S*) for the problem $O_2//C_{max}$. Similar to algorithm *GS*, algorithm *S* constructs an optimal schedule based on a flow shop type schedule with one job absent.

A number of authors considered the problem $O_2//C_{max}$ with various assumptions on the machine setting and job processing constraints. Adiri and Amit (1983) considered the problem with route-dependent processing times and introduced an $O(n)$-time algorithm to solve a special case with one dominant machine. Strusevich et al. (1999) tackled the same problem and proposed a 3/2-approximation algorithm. Breit et al. (2001) studied the problem where one machine is unavailable to process jobs for a certain time interval and introduced a 4/3-approximation algorithm. Mosheiov et al. (2018) examined the case in which a single maintenance has to be done on one machine and starts within a given time interval and developed a 3/2-approximation algorithm. Chernykh et al. (2013) explored the problem where machines and jobs are located at nodes of a network. Instead of delivering jobs to machines for processing, machines move through the network to process jobs. Initially, two machines are placed at the same node and must go back to this node after processing all the jobs. For this problem, the authors presented a 13/8-approximation algorithm. Khramova & Chernykh (2021) examined a similar yet even harder problem where the common initial node of the machines is not fixed but has to be chosen. Tellache et al. (2019) addressed the problem under resource constraints where a job can simultaneously visit both machines if certain conditions are met. The authors examined some special cases and proposed polynomial time algorithms. Huang et al. (2019) conducted a survey of intelligent algorithms including genetic algorithm, particle swarm optimization, cuckoo search algorithm and ant colony optimization for solving the problem $O_m//C_{max}$. The results showed that the cuckoo search algorithm is the best one to solve large scale problems. Recently, Mejía and Yuraszeck (2020) studied the problem $O_m//C_{max}$ with travel times between machines and sequence-dependent setup times. A variable neighborhood search algorithm which incorporates a new decoding scheme was proposed to solve the problem under consideration. The reader is referred to Kubiak (2022), Strusevich (2022) and Abreu et al. (2022) for an in-depth discussion of new and recent research on the open shop problem $O_m//C_{max}$.

In the literature, there is very limited research on the open shop scheduling problem with variable processing times, even for the two-machine case. Cheng and Shakhlevich (2007) dealt with the problem for which processing times are manageable and can be reduced with extra costs. They developed an algorithm with $O(n)$-time complexity for the problem to minimize the reduction cost for a given upper bound on the makespan and an $O(n\log n)$-time algorithm to simultaneously minimize the reduction cost and makespan. Kononov (1996) and Mosheiov (2002) independently showed that the two-machine open shop

scheduling problem with proportionally deteriorating jobs, $O_2/p_{ij}=b_{ij}t/C_{max}$, can be solved in $O(n)$ time by a modification of the algorithm *GS*. Gawiejnowicz and Kolinska (2021) considered the same problem and proposed an $O(n)$-time dispatching rule. This rule, called the Largest Alternate Deterioration Rate (*LADR*) first rule, works just as the *LAPT* rule by Pinedo and Schrage (1982) except that deterioration rates (b_{ij}) take the place of processing times (p_{ij}). Li (2011) examined a variant of this problem where one machine, called non-bottleneck machine, has unlimited capacity. That is, this non-bottleneck machine can simultaneously process an arbitrary quantity of jobs. The author developed an algorithm with pseudopolynomial time complexity for a specific case of this problem. If the deterioration of processing times is changed from a proportional form into a linear form, Kononov and Gawiejnowicz (2001) showed that the problem, $O_2/p_{ij}=a_{ij}+b_{ij}t/C_{max}$, is NP-hard, where a_{ij} represents the basic time demanded by machine M_i to process job j.

Note that the problem $O_2/p_{ij}=b_{ij}t/C_{max}$ is currently the only open shop scheduling problem with deteriorating jobs which has been shown to be polynomially solvable. The problem with three machines $O_3/p_{ij}=b_{ij}t/C_{max}$ becomes NP-hard (Kononov, 1996; Mosheiov, 2002). The reader is referred to Gawiejnowicz (2020a, 2020b) for more thorough reviews of the research on different time-related open shop models.

3. Methodology

As mentioned above, there are four different algorithms in literature for solving the traditional two-machine open shop scheduling problem $O_2//C_{max}$, namely algorithm *GS*, *LAPT* rule, algorithm *W* and algorithm *S*. All these algorithms have the same time complexity, $O(n)$. Among these algorithms, two of them, algorithm *GS* and *LAPT* rule, have been revised to solve the two-machine open shop scheduling problem with proportionally deteriorating jobs, $O_2/p_{ij}=b_{ij}t/C_{max}$ (Mosheiov, 2002; Gawiejnowicz and Kolinska, 2021). In this manuscript, we modify the other two algorithms, algorithm *W* and algorithm *S*, to solve the same problem. For completeness, algorithm *W* and algorithm *S* are concisely described in the following. For ease of expression, for each job j, let $\alpha_j = p_{1j}$ and $\beta_j = p_{2j}$, j=1, 2,…, n.

***Algorithm W*:**

It contains two phases. In phase 1, the set of jobs N is divided into three batches so that certain conditions are met. These three batches are then relabeled appropriately in phase 2 to ensure that optimality conditions hold.

Let $T=max\{\sum_{j=1}^{n}\alpha_j, \sum_{j=1}^{n}\beta_j\}$. Assume that $\alpha_j + \beta_j \le T$ for each job j=1, 2,…, n.

Phase 1: generating three batches

1. Find the job k such that
$$\sum_{j=1}^{k-1}(\alpha_j + \beta_j) \le T \text{ and } \sum_{j=1}^{k}(\alpha_j + \beta_j) > T.$$
2. Generate three batches as follows:

$$Q_1' = \{1, 2, \dots, k-1\}$$
$$Q_2' = \{k\}$$
$$Q_3' = \{k+1, k+2, \dots, n\}$$

Note that $Q_1' = \emptyset$ if k=1 and $Q_3' = \emptyset$ if k=n.

Phase 2: relabeling the batches

For any set of jobs Q, define $\alpha(Q)=\sum_{j\in Q}\alpha_j$ and $\beta(Q)=\sum_{j\in Q}\beta_j$. Assume that $\alpha(\emptyset)=\beta(\emptyset)=0$.

1. If the condition
$$\alpha(Q_1') \ge \beta(Q_2') \quad (1)$$
is satisfied, go to step 2. Else, go to step 3.

2. If the condition
$$\beta(Q_1') \ge \alpha(Q_3') \quad (2)$$
is satisfied, define $Q_1 = Q_1', Q_2 = Q_2', Q_3 = Q_3'$.
Else, define $Q_1 = Q_3', Q_2 = Q_1', Q_3 = Q_2'$.

3. If condition (2) holds, define $Q_1 = Q_2', Q_2 = Q_3', Q_3 = Q_1'$.
Else, go to step 4.

4. If the condition
$$\beta(Q_3') \ge \alpha(Q_2') \quad (3)$$
is satisfied, define $Q_1 = Q_3', Q_2 = Q_1', Q_3 = Q_2'$.
Else, define $Q_1 = Q_2', Q_2 = Q_3', Q_3 = Q_1'$.

The three-batch schedule is constructed as follows. Machine M_1 handles the batches in the sequence (Q_1, Q_2, Q_3) and machine M_2 handles the batches in the sequence (Q_2, Q_3, Q_1). The jobs in batch Q_1 first visit machine M_1, and then machine M_2, while the jobs in batches Q_2 and Q_3 first visit machine M_2, and then machine M_1. Note that the jobs within each batch can be processed in a random sequence without idle time in between and there is no unforced idle time between two successive batches.

***Algorithm S*:**

Algorithm *S* is also a two-phase method. A flow shop permutation schedule with n-1 jobs, whose makespan is no greater than the lower bound T, is found in phase 1. An optimal open shop schedule with n jobs is then constructed in phase 2.

It is well known that given a flow shop permutation schedule S where jobs are handled in the order 1, 2,…, n, its makespan is given by

$$C_{max}(S) = max_{1\le k\le n}\{\sum_{j=1}^{k}\alpha_j + \sum_{j=k}^{n}\beta_j\} = \sum_{j=1}^{k'}\alpha_j + \sum_{j=k'}^{n}\beta_j,$$

where job k' is called the critical job.

Phase 1: finding an optimal flow shop permutation schedule S' with n-1 jobs

1. Set $S_1' = \{1,2,\dots,n-1\}$ and compute $C_{max}(S_1')$. Let $k(1)$ be the critical job.
2. If $C_{max}(S_1') \le T$, stop and output S_1'. Else, set q=2 and go to step 3.

3. Set $S'_q = \{q, q+1, \ldots, q+n-2\}$. If $q+n-2>n$, *set* $q+j=q+j-n$ *for all* $j>n-q$.
If $q=k(1)+1$, stop and output S'_q. Otherwise, compute

$$C_{max}(S'_q) = C_{max}(S'_{q-1}) - \alpha_{q-1} + \beta_{q+n-2}.$$

If $C_{max}(S'_q) \le T$, stop and output S'_q. Else, set $q=q+1$ and go to step 3.

Phase 2: finding an optimal open shop schedule S with n jobs
Let S' be the schedule output by phase 1, and r be the omitted job. Construct an open shop schedule S as follows. Machine M_1 processes the set of jobs $N\backslash\{r\}$ first in the same order as in schedule S', followed by job r. Machine M_2 processes job r first, then all jobs in $N\backslash\{r\}$ in the same order as on machine M_1. The set of jobs $N\backslash\{r\}$ first visit machine M_1, then machine M_2, while job r first visits machine M_2, then machine M_1.

In the following, we propose the modified algorithm W (called algorithm W_m) and the modified algorithm S (called algorithm S_m) to solve the problem $O_2/p_{ij}=b_{ij}t/C_{max}$. It is seen that both algorithms obtain an optimal schedule in $O(n)$ time. Without loss of generality, let $t_0 = 1$. For ease of expression, for each job j, let $\mu_j = b_{1j}$ and $\nu_j = b_{2j}$, j=1, 2, …, n.

As shown by Mosheiov (2002), a lower bound T on the optimal makespan of the problem $O_2/p_{ij}=b_{ij}t/C_{max}$ is given by

$$\mathrm{T} = \max\{\textstyle\prod_{\mathrm{j}=1}^{\mathrm{n}}\left(1+\mu_{\mathrm{j}}\right), \prod_{\mathrm{j}=1}^{\mathrm{n}}(1+\nu_{\mathrm{j}})\}. \qquad (4)$$

We assume that for each job j, $\left(1+\mu_j\right)\left(1+\nu_j\right) \le T$. Otherwise, a schedule with makespan $\left(1+\mu_j\right)\left(1+\nu_j\right)$ for some job j can be obtained as shown in Fig. 1. Since the value $\left(1+\mu_j\right)\left(1+\nu_j\right)$ is obviously a lower bound on the optimal makespan, this schedule is optimal.

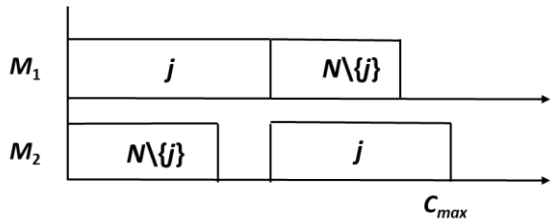

Fig.1. A trivial optimal schedule.

3.1. Algorithm W_m

Algorithm W_m consists of the following two phases. Phase 1 splits the set of jobs N into three batches and phase 2 properly relabels the generated batches. It is easily seen that algorithm W_m runs in $O(n)$ time.

Phase 1: generating three batches
1. Find the job k such that

$$\textstyle\prod_{\mathrm{j}=1}^{\mathrm{k}-1}(1+\mu_{\mathrm{j}})(1+\nu_{\mathrm{j}}) \le \mathrm{T} \text{ and}$$
$$\textstyle\prod_{\mathrm{j}=1}^{\mathrm{k}}(1+\mu_{\mathrm{j}})(1+\nu_{\mathrm{j}}) > \mathrm{T}.$$

2. Same as the step 2 in phase 1 of algorithm *W.*

Phase *2*: relabeling the batches
For any set of jobs Q, let $\mu(Q)=\prod_{j\in Q}(1+\mu_j)$ and $\nu(Q)=\prod_{j\in Q}(1+\nu_j)$. Define $\mu(\emptyset)=\nu(\emptyset)=0$.

1. If the condition
$\mu(Q'_1) \ge \nu(Q'_2)$ *(5)*
is satisfied, go to step 2. Else, go to step 3.
2. If the condition
$\nu(Q'_1) \ge \mu(Q'_3)$ *(6)*
is satisfied, define $\mathrm{Q}_1 = \mathrm{Q}'_1, \mathrm{Q}_2 = \mathrm{Q}'_2, \mathrm{Q}_3 = \mathrm{Q}'_3$.
Else, define $Q_1 = Q'_3, Q_2 = Q'_1, Q_3 = Q'_2$.
3. If condition (6) holds, define $\mathrm{Q}_1 = \mathrm{Q}'_2, \mathrm{Q}_2 = \mathrm{Q}'_3, \mathrm{Q}_3 = \mathrm{Q}'_1$.
Else, go to step 4.
4. If the condition
$\nu(Q'_3) \ge \mu(Q'_2)$ *(7)*
is satisfied, define $\mathrm{Q}_1 = \mathrm{Q}'_3, \mathrm{Q}_2 = \mathrm{Q}'_1, \mathrm{Q}_3 = \mathrm{Q}'_2$.
Else, define $Q_1 = Q'_2, Q_2 = Q'_3, Q_3 = Q'_1$.

Without loss of generality, let
$T = \mu(N) = \prod_{j=1}^{n}\left(1+\mu_j\right) = \prod_{j=1}^{n}\left(1+\nu_j\right) = \nu(N)$.
That is, the workloads on both machines are equal. A dummy job can be introduced if necessary. More specifically, if $\mu(N) > \nu(N)$, a dummy job j with $\mu_j = 0$ and $\nu_j = \frac{\mu(N)}{\nu(N)} - 1$ is introduced. Otherwise, if $\nu(N) > \mu(N)$, a dummy job j with $\nu_j = 0$ and $\mu_j = \frac{\nu(N)}{\mu(N)} - 1$ is introduced.

Lemma 1. For the problem $O_2/p_{ij}=b_{ij}t/C_{max}$, if there exist three batches Q_1, Q_2 and Q_3 of jobs such that the following inequalities hold,

$$\mu(Q_1) \ge \nu(Q_2) \qquad (8)$$

$$\nu(Q_1) \ge \mu(Q_3) \qquad (9)$$

$$\mu(Q_1)\,\nu(Q_1) \le T \qquad (10)$$

then an optimal schedule S with makespan T can be generated by processing the batches in the order (Q_1, Q_2, Q_3) on machine M_1 and the order (Q_2, Q_3, Q_1) on machine M_2.

Proof: We remark that in schedule S the jobs in batch Q_1 first visit machine M_1, and then machine M_2, while the jobs in batches Q_2 and Q_3 first visit machine M_2, and then machine M_1. Fig. 2 shows the four possible structures of schedule S. Since its makespan has the same value as the lower bound T, schedule S is clearly optimal. Now, it is sufficient to show that in schedule S the processing of each job on different machines do not overlap. That is, schedule S is feasible. Let S_{il} and C_{il}, respectively, be the beginning time and finishing time of the processing of batch Q_l on machine i, i=1, 2; l=1, 2, 3. We need to show that $C_{22} \le S_{12}$, $C_{23} \le S_{13}$ and $C_{11} \le S_{21}$.

If condition (8) is satisfied, we have

$$C_{22} = \nu(Q_2) \le \mu(Q_1) = S_{12}.$$

If condition (9) holds, then

$$C_{23} = \nu(Q_2)\nu(Q_3) = \frac{T}{\nu(Q_1)} \le \frac{T}{\mu(Q_3)} = \mu(Q_1)\mu(Q_2) = C_{12} \le S_{13}$$

Due to inequality (10) we have

$$C_{11} = \mu(Q_1) \le \frac{T}{\nu(Q_1)} = \nu(Q_2)\nu(Q_3) = C_{23} \le S_{21}.$$

Therefore, the schedule *S* is feasible and hence optimal.

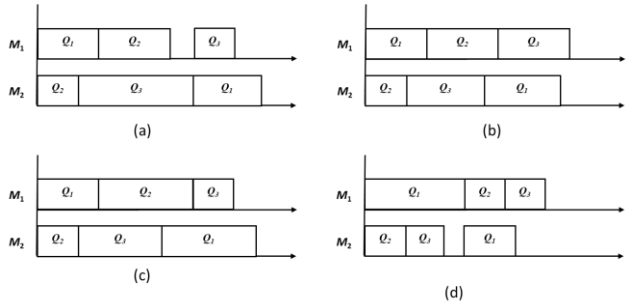

Fig. 2. Four possible structures of an optimal schedule *S*.

Lemma 2. For each batch Q_l, l=1, 2, 3, generated by algorithm *W_m*, the following condition holds.

$$\mu(Q_l)\nu(Q_l) \le T \quad (11)$$

Proof: Since $T = \mu(N) = \nu(N)$, we have

$$\mu(N)\nu(N) > T. \quad (12)$$

The job k in step 1 of phase 1 exists, and hence inequality (11) holds for l=1. As we assume for each job j, $\left(1+\mu_j\right)\left(1+\nu_j\right) \le T$, j=1, 2, …, n, inequality (11) holds for l=2. Finally, since

$\prod_{j=1}^{k}(1+\mu_j)(1+\nu_j) = \mu(Q_1)\mu(Q_2)\nu(Q_1)\nu(Q_2) > T$,

due to inequality (12) we have that $\mu(Q_3)\nu(Q_3) \le T$. Thus, inequality (11) holds for l=3. □

Theorem 1. The three batches Q_1, Q_2 and Q_3 generated by algorithm *W_m* satisfy inequalities (8)-(10).

Proof: The result of Lemma 2 guarantees that condition (10) is satisfied. If both conditions (5) and (6) are satisfied, then conditions (8) and (9) are satisfied. There are four cases to be considered.

Case 1: inequality (5) is valid and inequality (6) is not valid

We have $\mu(Q_1') \ge \nu(Q_2')$ and $\mu(Q_3') > \nu(Q_1')$. Since $T = \mu(N) = \nu(N)$, it follows that $\mu(Q_2') \le \nu(Q_3')$. Hence, conditions (8) and (9) hold for $Q_1 = Q_3', Q_2 = Q_1', Q_3 = Q_2'$.

Case 2: inequality (5) is not valid and inequality (6) is valid

We have $\mu(Q_1') < \nu(Q_2')$ and $\mu(Q_3') \le \nu(Q_1')$. Since $T = \mu(N) = \nu(N)$, it follows that $\mu(Q_2') \ge \nu(Q_3')$. This implies that conditions (8) and (9) are satisfied for $Q_1 = Q_2', Q_2 = Q_3', Q_3 = Q_1'$.

Case 3: both inequalities (5) and (6) are not valid and inequality (7) is valid

As inequality (5) is not valid and inequality (7) is valid, we have $\mu(Q_1') < \nu(Q_2')$ and $\mu(Q_2') \le \nu(Q_3')$. Similarly, since $T = \mu(N) = \nu(N)$, it follows that $\mu(Q_3') \ge \nu(Q_1')$. This result together with inequality (7) implies that conditions (8) and (9) hold for $Q_1 = Q_3', Q_2 = Q_1', Q_3 = Q_2'$.

Case 4: all the inequalities (5), (6) and (7) are not valid

As both inequalities (5) and (7) are not valid, we have $\mu(Q_1') < \nu(Q_2')$ and $\nu(Q_3') < \mu(Q_2')$. It is seen that conditions (8) and (9) are satisfied for $Q_1 = Q_2', Q_2 = Q_3', Q_3 = Q_1'$.

Example 1: A problem instance of $O_2/p_{ij}=b_{ij}t/C_{max}$.

j	1	2	3	4	5
$b_{1j}=\mu_j$	2	3	1	2	4
$b_{2j}=\nu_j$	4	1	2	3	1

By definition, the lower bound

$T = max\{\prod_{j=1}^{n}(1+\mu_j), \prod_{j=1}^{n}(1+\nu_j)\}$

$= max\{3\times4\times2\times3\times5, 5\times2\times3\times4\times2\} = 360.$

Phase 1:

1. Since $\prod_{j=1}^{2}(1+\mu_j)(1+\nu_j) = 3\times5\times4\times2 = 120 \le T$ and

$\prod_{j=1}^{3}(1+\mu_j)(1+\nu_j) = 3\times5\times4\times2\times2\times3 = 720 > T$,

it follows that k=3.

2. By construction, $Q_1' = \{1,2\}, Q_2' = \{3\}$, and $Q_3' = \{4,5\}$.

Phase 2:

As $\mu(Q_1') = 3\times4 = 12 \ge \nu(Q_2') = 3$ and

$\nu(Q_1') = 5\times2 = 10 < \mu(Q_3') = 3\times5 = 15$,

define $Q_1 = Q_3' = \{4,5\}, Q_2 = Q_1' = \{1,2\}, Q_3 = Q_2' = \{3\}$.

The schedule generated by algorithm *W_m* is shown in figure 3. As its makespan is the same as the lower bound value *T*, the schedule is obviously optimal.

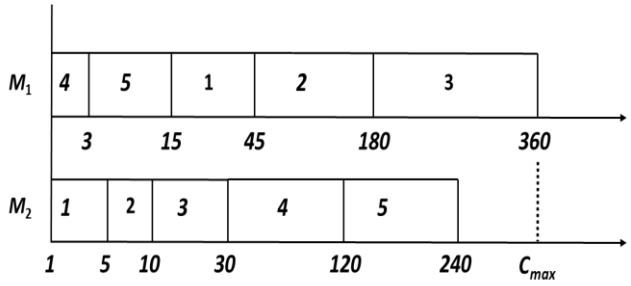

Fig. 3.Schedule obtained by algorithm *W_m* for instance in example 1.

3.2. Algorithm S_m

Given a flow shop permutation schedule S where both machines process jobs in the order 1, 2,…, n, its makespan is given by

$$C_{max}(S) = \max_{1\le k\le n}\{\prod_{j=1}^{k}(1+\mu_j)\prod_{j=k}^{n}(1+v_j)\}$$

$$= \prod_{j=1}^{k'}\left(1+\mu_j\right)\prod_{j=k'}^{n}(1+v_j) \quad (13)$$

where job k' is called the critical job of schedule S. For an arbitrary value of q, $1 \le q \le n$, define by $S'_q = \{q, q+1, \dots, q+n-2\}$ the flow shop permutation schedule where jobs are handled in the order $q, q+1, \dots, q+n-2$. In the following, by an abuse of notation, we set $\mu_j = \mu_{j-n}$ and $v_j = v_{j-n}$ for all $j>n$. Algorithm *S_m* consists of the following two phases. It is apparent the algorithm has an $O(n)$ time complexity.

Phase 1: finding an optimal flow shop permutation schedule S' with n-1 jobs

1. Set $S'_1 = \{1,2,\dots,n-1\}$ and compute

$$C_{max}(S'_1) = \max_{1\le k\le n-1}\{\prod_{j=1}^{k}(1+\mu_j)\prod_{j=k}^{n-1}(1+v_j)\}$$

$$= \prod_{j=1}^{k(1)}(1+\mu_j)\prod_{j=k(1)}^{n-1}(1+v_j)$$

where $k(1)$ is the critical job of schedule S'_1.

2. If $C_{max}(S'_1) \le T$, stop and output S'_1. Else, set q=2 and go to step 3.

3. Set $S'_q = \{q, q+1, \dots, q+n-2\}$.

If $q=k(1)+1$, stop and output S'_q. Else, compute

$$C_{max}(S'_q) = C_{max}(S'_{q-1}) * (1+v_{q+n-2})/\left(1+\mu_{q-1}\right).$$

If $C_{max}(S'_q) \le T$, stop and output S'_t. Else, set $q=q+1$ and go to step 3.

Phase 2: finding an optimal open shop schedule S with n jobs

Same as phase 2 of algorithm *S*.

Lemma 1. Consider a permutation flow sop schedule $S'_{q-1} = \{q-1, q, \dots, q+n-3\}$ with $C_{max}(S'_{q-1}) > T$ in phase 1 of algorithm *S_m*. Let $k(q-1)$ be the critical job of schedule S'_{q-1}. If $q \le k(q-1)$, then $C_{max}(S'_q) = C_{max}(S'_{q-1}) * (1+v_{q+n-2})/\left(1+\mu_{q-1}\right)$, where $S'_q = \{q, q+1, \dots, q+n-2\}$.

Proof: Note that if a critical job for schedule S'_q is the same as that of schedule S'_{q-1}, i.e., $k(q)=k(q-1)$, we have

$$C_{max}(S'_q) = \prod_{j=q}^{k(q-1)}(1+\mu_j)\prod_{j=k(q-1)}^{q+n-2}(1+v_j)$$

$$= \prod_{j=q-1}^{k(q-1)}(1+\mu_j)\prod_{j=k(q-1)}^{q+n-3}(1+v_j) * (1+v_{q+n-2})/\left(1+\mu_{q-1}\right)$$

$$= C_{max}(S'_{q-1}) * (1+v_{q+n-2})/\left(1+\mu_{q-1}\right)$$

Hence, it is sufficient to show that a critical job for schedule S'_q is the same as that for schedule S'_{q-1}, for each $q \le k(q-1)$. Considering schedule S'_{q-1}, we have

$$C_{max}(S'_{q-1}) = \prod_{j=q-1}^{k(q-1)}(1+\mu_j)\prod_{j=k(q-1)}^{q+n-3}(1+v_j).$$

If $\prod_{j=k(q-1)}^{q+n-3}(1+v_j) \le \prod_{j=k(q-1)}^{q+n-3}\left(1+\mu_{j+1}\right)$, then $C_{max}(S'_{q-1}) \le \prod_{j=q-1}^{k(q-1)}(1+\mu_j)\prod_{j=k(q-1)}^{q+n-3}(1+\mu_{j+1}) = \prod_{j=q-1}^{q+n-2}(1+\mu_j) = \prod_{j=1}^{n}(1+\mu_j) \le T$

which is contradictory to the assumption $C_{max}(S'_{q-1}) > T$. Thus, it follows that

$$\prod_{j=k(q-1)}^{q+n-3}(1+v_j) > \prod_{j=k(q-1)}^{q+n-3}\left(1+\mu_{j+1}\right) \quad (14)$$

As shown in Fig. 4, by construction we must have

$$\prod_{j=l}^{k(q-1)}\left(1+\mu_j\right) \ge \prod_{j=l}^{k(q-1)}(1+v_{j-1}), \text{ for } l=q,\ q+1,\dots,\ k(q-1) \quad (15)$$

Inequality (15) with $l=q+1$ together with inequality (14) implies that a critical job of schedule S'_q is the same as that of schedule S'_{q-1}, i.e., $k(q)=k(q-1)$. □

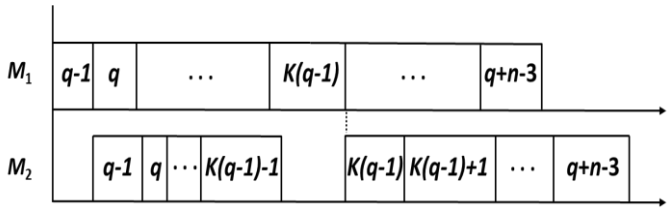

Fig. 4. A permutation flow shop schedule S'_{q-1}.

Lemma 2. Phase 1 of algorithm *S_m* finds a schedule S' such that $C_{max}(S') \le T$ for a permutation flow shop with n-1 jobs.

Proof: Consider the permutation flow shop schedules $S'_q = \{q, q+1, \dots, q+n-2\}$, q=1, 2,…, $k(1)$. If $C_{max}(S'_1) \le T$, we are done. Otherwise, as shown in lemma 1, we can set $k(q)=k(1)$ provided that $C_{max}(S'_q) > T$ for each $q \le k(1)$. If there is a value $q \le k(1)$ such that $C_{max}(S'_q) \le T$,we are done. Otherwise, we need to show that $C_{max}(S'_q) \le T$ for $q=k(1)+1$, i.e., $C_{max}(S'_{k(1)+1}) \le T$.

Set $q=k(1)+1$ and $k(q-1)=k(1)$. Considering the schedule S'_{q-1} as shown in Fig. 4 with job index $k(q-1)$ replaced by $k(1)$, it follows that

$\prod_{j=k(1)}^{l}\left(1+\mu_{j+1}\right) \le \prod_{j=k(1)}^{l}(1+\nu_j)$
for $l=k(1), k(1)+1,\ldots,q+n-3$. (16)
Since $C_{max}(S'_{q-1}) > T$, inequality (16) holds. That is,
$\prod_{j=k(q-1)}^{q+n-3}\left(1+\mu_{j+1}\right) =$
$\prod_{j=k(1)}^{k(1)+n-2}\left(1+\mu_{j+1}\right) < \prod_{j=k(1)}^{q+n-3}(1+\nu_j) =$
$\prod_{j=k(1)}^{k(1)+n-2}(1+\nu_j)$ (17)
Combining inequalities (16) and (17), we get
$\prod_{j=k(1)}^{l}\left(1+\mu_{j+1}\right) \le \prod_{j=k(1)}^{l}(1+\nu_j)$
for $l=k(1), k(1)+1,\ldots,q+n-2$. (18)
Inequality (18) can be further rewritten as
$\prod_{j=k(1)+1}^{l}\left(1+\mu_j\right) \le \prod_{j=k(1)}^{l-1}(1+\nu_j)$
for $l=k(1)+1, k(1)+2,\ldots,k(1)+n$. (19)
As $S'_{k(1)+1} = \{k(1)+1, k(1)+2, \ldots, k(1)+n-1\}$, a critical job of schedule $S'_{k(1)+1}$ can happen at $k(1)+1$, $k(1)+2,\ldots,k(1)+n-1$. Suppose a critical job of schedule $S'_{k(1)+1}$ occurs at $k(1)+h$, where $1 \le h \le n-1$. By construction, we have
$C_{\max}(S'_{k(1)+1}) = \prod_{j=k(1)+1}^{k(1)+h}(1+\mu_j)\prod_{j=k(1)+h}^{k(1)+n-1}(1+\nu_j)$
Inequality (19) with $l=k(1)+h$, where $1 \le h \le n-1$, implies that
$C_{max}(S'_{k(1)+1}) \le \prod_{j=k(1)}^{k(1)+h-1}(1+\nu_j)\prod_{j=k(1)+h}^{k(1)+n-1}(1+\nu_j)$
$= \prod_{j=k(1)}^{k(1)+n-1}(1+\nu_j) = \prod_{j=1}^{n}(1+\nu_j) \le T$
Thus, a permutation flow shop schedule S' with $C_{max}(S') \le T$ must be found by Phase 1 of algorithm *S_m*. □

Lemma 3. Phase 2 of algorithm *S_m* finds an open shop schedule *S* with *n* jobs such that $C_{max}(S) \le T$.

Proof: Phase 2 of algorithm *S_m* constructs an open shop schedule *S* with *n* jobs from the flow shop permutation schedule S' with *n*-1 jobs obtained in phase 1 of algorithm *S_m*. Let *r* be the job in schedule *S* but not in schedule S'. Given a schedule *S*, let $C_i(S)$ be the finishing time of the final job processed by machine M_i, i=1, 2.
By construction, all the *n* jobs are continuously processed by machine M_1. Thus, $C_1(S) = \prod_{j=1}^{n}(1+\mu_j) \le T$. Let $I_2(S')$ be the total idle time of machine M_2 in schedule S'. As all the jobs in schedule S' first visit machine M_1, then machine M_2, the processing of jobs on machine M_2 can be arbitrarily delayed if needed. Since in schedule *S* machine M_2 processes job *r* first, the processing time of job *r* is equal to v_r. If $I_2(S') \ge v_r$, i.e., the total idle time is no less than the time needed to handle job *r* on machine M_2, we have $C_2(S) = C_{max}(S') \le T$. Otherwise, the processing of some of the jobs in S' on machine M_2 will be delayed due to the handle of job *r* on machine M_2 and the resulted schedule *S* will have no idle time on machine M_2. In this case, we have $C_2(S) = \prod_{j=1}^{n}(1+\nu_j) \le T$. □

Theorem 2. Algorithm *S_m* finds an optimal schedule for the problem $O_2/p_{ij}=b_{ij}t/C_{max}$.

Proof: It is easily seen that the open shop schedule generated by algorithm *S_m* has no overlapping jobs and hence is feasible. Since its makespan, by lemma 3, is less than or equal to *T*, a lower bound on the optimal makespan, the open shop schedule generated by algorithm *S_m* is optimal for the problem $O_2/p_{ij}=b_{ij}t/C_{max}$. □

Example 2: Consider again the problem instance in example 1.

Phase 1:
1. Set S'_1={1, 2, 3, 4} and compute $C_{max}(S'_1)$=max{360, 288, 288, 288}=360 with $k(1)$=1.
2. As $C_{max}(S'_1) \le T$=360, output schedule S'_1.

Phase 2:
The omitted job is job 5 (r=5). Machine M_1 processes the set of jobs {1, 2, 3, 4} in the order 1, 2, 3, 4, followed by job 5. Machine M_2 processes job 5 first, then the set of jobs {1, 2, 3, 4} in the order 1, 2, 3, 4. Jobs 1, 2, 3 and 4 first visit machine M_1 and then machine M_2, while job 5 first visit machine M_2 and then machine M_1. The resulted schedule is shown in Fig. 5. Given that its makespan is the same as the lower bound value *T*, the schedule is clearly optimal.

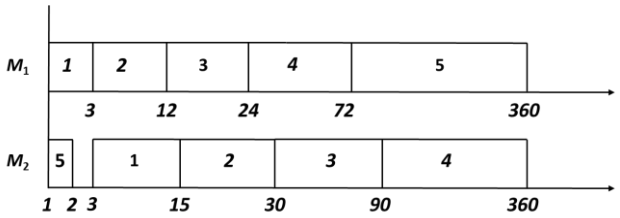

Fig. 5.Schedule obtained by algorithm *S_m* for instance in example 1.

4. Discussion

Another problem which can be viewed as a generalization of the one examined in this manuscript is the open shop scheduling problem with two machines and proportionally-linear deteriorating jobs, $O_2/p_{ij}=b_{ij}(a+ct)/C_{max}$, where $a \ge 0$ and $c \ge 0$. The problem reduces to $O_2/p_{ij}=b_{ij}t/C_{max}$ when a=0 and c=1. We remark that both the algorithms proposed in section 3 for the problem $O_2/p_{ij}=b_{ij}t/C_{max}$ can be slightly modified to solve the problem $O_2/p_{ij}=b_{ij}(a+ct)/C_{max}$. The proof is straightforward and hence is omitted.
Again, assume that $t_0 = 1$. Based on the results by Kononov (1998), a lower bound on the optimal makespan for the problem $O_2/p_{ij}=b_{ij}(a+ct)/C_{max}$ is given by

$$T = (1+\frac{a}{c})\max\{\prod_{j=1}^{n}\left(1+c\mu_j\right), \prod_{j=1}^{n}(1+c\nu_j)\} - \frac{a}{c} \quad (20)$$

Here, the trivial case where the optimal makespan is equivalent to the total processing time of a single job is excluded. Equality (20) is a counterpart of equality (4) for problem $O_2/p_{ij}=b_{ij}t/C_{max}$, where
$(1+\frac{a}{c})\prod_{j=1}^{n}\left(1+c\mu_j\right) - \frac{a}{c}$ is the workload of machine M_1 and $(1+\frac{a}{c})\prod_{j=1}^{n}(1+c\nu_j) - \frac{a}{c}$ is the workload of machine M_2. For ease of execution, let $T' = max\{\prod_{j=1}^{n}\left(1+c\mu_j\right), \prod_{j=1}^{n}(1+c\nu_j)\}$. The modified algorithms, named algorithm *W_m*2 and

algorithm *S_m*2, for solving the problem $O_2/p_{ij}=b_{ij}(a+ct)/C_{max}$ are given as follows:

Algorithm *W_m*2：

Phase 1: generating three batches

1. Find the job *k* such that
   $\prod_{j=1}^{k-1}(1+c\mu_j)(1+c\nu_j) \le T'$ and
   $\prod_{j=1}^{k}(1+c\mu_j)(1+c\nu_j) > T'$.
2. Same as the step 2 in phase 1 of algorithm *W_m*.

Phase 2: relabeling the batches

Same as the phase 2 of algorithm *W_m* except that for any set of jobs *Q*, $\mu(Q)=\prod_{j\in Q}(1+c\mu_j)$ and $\nu(Q)=\prod_{j\in Q}(1+c\nu_j)$.

Algorithm *S_m*2：

Phase 1: finding an optimal flow shop permutation schedule S' with *n*-1 jobs

1. Set $S_1' = \{1,2,\dots,n-1\}$ and compute

$$C_{max}(S_1') = \max_{1\le k\le n-1}\{\prod_{j=1}^{k}(1+c\mu_j)\prod_{j=k}^{n-1}(1+c\nu_j)\}$$

$$= \prod_{j=1}^{k(1)}(1+c\mu_j)\prod_{j=k(1)}^{n-1}(1+c\nu_j)$$

where $k(1)$ is the critical job of schedule S_1'.

2. If $C_{max}(S_1') \le T'$, stop and output S_1'. Else, set q=2 and go to step 3.

3. Set $S_q' = \{q, q+1, \dots, q+n-2\}$.

If $q=k(1)+1$, stop and output S_t'. Else, compute

$$C_{max}(S_q') = C_{max}(S_{q-1}') * (1+c\nu_{q+n-2}) / (1+c\mu_{q-1}).$$

If $C_{max}(S_q') \le T'$, stop and output S_t'. Else, set q=q+1 and go to step 3.

Phase 2: finding an optimal open shop schedule *S* with *n* jobs

Same as phase 2 of algorithm *S_m*.

We remark that in these two algorithms only parameter *c* is used to resolve the order jobs are handled by each machine, while parameter *a* is used when the actual finishing time of every job is calculated. Obviously, both algorithms have *O*(*n*)-time complexity.

5. Conclusion

In this manuscript, we study the two-machine open shop scheduling problem with proportionally deteriorating processing times and the performance measure of minimizing makespan. Currently, there are two exact methods in literature with complexity *O*(*n*) for solving this problem. Two new algorithms with the same complexity *O(n)* are developed. This manuscript also proved theoretically that both algorithms proposed can optimally solve the problem $O_2/p_{ij}=b_{ij}t/C_{max}$. Two numerical examples are given to demonstrate the execution of these algorithms. Also, both algorithms are further revised to solve a more generalized problem where the time it takes to process a job is a general linear function of its beginning time. This manuscript contributes to some extent to the development of solution methods and theoretical properties for the problem $O_2/p_{ij}=b_{ij}t/C_{max}$.

For the open shop scheduling problem with two machines and makespan objective, this manuscript considers two basic models of deteriorating processing times, proportional deterioration and proportionally-linear deterioration. One possible direction for future research is to explore other different models of deteriorating processing times, for example linear deterioration, exponential deterioration, generally-nonlinear deterioration, etc. Another possible future research direction is to examine these problems with other objectives, such as total completion times and total tardiness. To the authors' knowledge, the two-machine open shop scheduling problem with makespan objective and shortening job processing times has never been studied. Hence, an appealing area for future research is to examine various models for shortening job processing times of the open shop scheduling problem with two machines and makespan objective.

References

Abreu, L.R., Prata, B.A., Framinan, J.M. & Nagano, M.S. (2022). New efficient heuristics for scheduling open shops with makespan minimization. *Computers & Operations Research*, 142 (2022) 105744. doi.:10.1016/j.cor.2022.105744.

Adiri, I. & Amit, N. (1983). Route-dependent open-shop scheduling. *IIE Transactions*, 15(3), 231-234.

Ahmadian, M.M., Khatami, M., Salehipour, A. & Cheng, T.C.E. (2021). Four decades of research on the open-shop scheduling problem to minimize the makespan. *European Journal of Operational Research*, 295, 399-426.

Alidaee, B. & Womer, N.K. (1999). Scheduling with time dependent processing times: Review and extensions. *Journal of the Operational Research Society*, 50, 711–720.

Behnamian, J. (2019). Diversified Particle Swarm Optimization for Hybrid Flowshop Scheduling. *Journal of Optimization in Industrial Engineering*, 12(2), 107- 119. doi: 10.22094/JOIE.2018.671.1433.

Breit, J., Schmidt, G. & Strusevich, V.A. (2001). Two-machine open shop scheduling with an availability constraint. *Operations Research Letters*, 29, 65-77.

Cheng, T.C.E. & Sharkhlevich, N,V. (2007). Two-machine open shop problem with controllable processing times. *Discrete Optimization*, 4, 175-184.

Cheng, T.C.E., Ding, Q. & Lin, B.M.-T. (2004). A concise survey of scheduling with time-dependent processing times. *European Journal of Operational Research*, 152, 1–13.

Chernykh, I., Kononov, A. & Sevastyanov, S. (2013). Efficient approximation algorithms for the routing open shop problem. *Computers and Operations Research*, 40(3), 841-847.

de Werra, D. (1989). Graph-theoretical models for preemptive scheduling. Advances in Project Scheduling (p171-185). *Elsevier*, Amsterdam, Netherlands.

Enayati, M., Asadi-Gangraj E. & Paydar, M.M. (2021). Scheduling on Flexible Flow Shop with Cost-related Objective Function Considering Outsourcing Options. *Journal of Optimization in Industrial Engineering*, 14(2), 53-72. doi: 10.22094/JOIE.2020.1873983.1674

Gawiejnowicz, S. & Kolinska, M. (2021). Two- and three-machine open shop scheduling using LAPT-like rules. *Computers & Industrial Engineering*, 157, (2021) 107261. doi: 10.1016/j.cie.2021.107261.

Gawiejnowicz, S. (1996a). A note on scheduling on a single processor with speed dependent on a number of executed jobs. *Information Processing Letters*, 57, 297–300.

Gawiejnowicz, S. (1996b). Brief survey of continuous models of scheduling. *Foundations of Computing and Decision Sciences*, 21, 81–100.

Gawiejnowicz, S. (2020a). A review of four decades of time-dependent scheduling: main results, new topics and open problems. *Journal of Scheduling*, 23, 3-47.

Gawiejnowicz, S. (2020b). Models and algorithms of time-dependent scheduling. *Springer*, Berlin-Heidelberg, Germany. doi:10.1007/978-3-662-59362-2.

Gawiejnowicz, S. (2008). Time-dependent scheduling. *Springer*, Berlin, Germany.

Gonzalez, T. & Sahni, S. (1976). Open shop scheduling to minimize finish time. *Journal of the Association for Computing Machinery*, 23, 665-679.

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

Gribkovskaia, I.V., Lee, C.-Y., Strusevich, V.A. & de Werra, D. (2006). Three is easy, two is hard: open shop sum-batch scheduling problem refined. *Operations Research Letters*, 34, 459-464.

Huang, Z., Zhuang, Z., Cao, Q., Lu, Z., Guo, L. & Qin, W. (2019). A survey of intelligent algorithms for open shop scheduling problem. Procedia CIRP 83 (2019) 569–574.

Khramova, A.P. & Chernykh, I. (2021). A new algorithm for the two-machine open shop and the polynomial solvability of a scheduling problem with routing. *Journal of Scheduling*, 24**,** 405–412. doi:10.1007/s10951-021-00694-7.

Kononov, A. & Gawiejnowicz, S. (2001). NP-hard cases in scheduling deteriorating jobs on dedicated machines. *Journal of the Operations Research Society*, 52, 708-718.

Kononov, A. (1996). Combinatorial complexity of scheduling jobs with simple linear deterioration. *Discrete Analysis and Operations Research*, 3, 15-32. (In Russian)

Kononov, A. (1998). Single machine scheduling problems with processing times proportional to an arbitrary function. *Discrete Analysis and Operations Research*, 5, 17-37. (In Russian)

Kubiak, W. (2022). A book of Open Shop Scheduling: Algorithms, Complexity and Applications. *Springer Nature*, Berlin, Germany. doi:10.1007/978-3-030-91025-9.

Li, S.-S. (2011). Scheduling proportionally deteriorating jobs in two-machine open shop with a non-bottleneck machine. *Asia-Pacific Journal of Operations Research*, 28, 623-631.

Mejía G. & Yuraszeck, F. (2020). A self-tuning variable neighborhood search algorithm and an effective decoding scheme for open shop scheduling problems with

travel/setup times. *European Journal of Operational Research*, 285, 484-496.

Mosheiov, G. (2002). Complexity analysis of job-shop scheduling with deteriorating jobs. *Discrete Applied Mathematics*, 117, 195-209.

Mosheiov, G., Sarig, A., Strusevich, V.A. & Mosheiff, J. (2018). Two-machine flow shop and open shop scheduling problems with a single maintenance window. *European Journal of Operational Research*, 271, 388-400.

Pinedo, M. & Schrage, L. (1982). Stochastic shop scheduling: A survey. Deterministic and Stochastic Scheduling (p181-196). *Springer*, Berlin, Germany.

Soper, A.J. (2015). A cyclical search for the two machine flow shop and open shop to minimise finishing time. *Journal of Scheduling*, 18(3), 311-314.

Strusevich, V.A. & Rustogi, K. (2017). Scheduling with time-changing effects and rate-modifying activities. *Springer*, Berlin, Germany.

Strusevich, V.A., Van De Waart, A. & Dekker, R. (1999). A 3/2 algorithm for two-machine open shop with route-dependent processing times. *Journal of Heuristics*, 5(1), 5-28.

Strusevich, V.A. (2022). Complexity and approximation of open shop scheduling to minimize the makespan: A review of models and approaches. *Computers & Operations Research*, 144 (2022) 105732. doi: 10.1016/j.cor.2022.105732.

Tellache, N.E., Boudhar, M. & Yalaoui, F. (2019). Two-machine open shop problem with agreement graph. *Theoretical Computer Science*, 796, 154-168.

Yavari, S., Azab, A., Baki, M.F., Alcelay, M. & Britt, J. (2020). Machine Scheduling for Multitask Machining. *Journal of Optimization in Industrial Engineering*, 13(2), 1-15. doi: 10.22094/JOIE.2020.1869865.1660

This article can be cited: Liaw, C. (2022). Two-Machine Open Shop Scheduling with Proportionally Deteriorating Jobs and Makespan Objective. *Journal of Optimization in Industrial Engineering*, *15*(2), 313-321. Doi: 10.22094/joie.2022.1949101.1926