HYBRID TWO STAGE FLOWSHOP SCHEDULING WITH SECONDARY RESOURCES BASED ON TIME BUCKETS

Alex J. Ruiz-Torres
Departamento de Gerencia, Universidad de Puerto Rico, USA
alex.ruiztorres@upr.edu
https://orcid.org/0000-0002-7528-236X

Giuseppe Paletta
Dipartimento di Economia, Universita` della Calabria, Italy
giuseppe.paletta@unical.it
https://orcid.org/0000-0001-8446-781X

Belarmino Adenso-Díaz
Departmento de Ingenieria Industrial, Universidad de Oviedo, Spain,
adenso@uniovi.es
https://orcid.org/0000-0002-6519-9810

Publicado en **INTERNATIONAL JOURNAL OF PRODUCTION RESEARCH**, Vol. 60, No. 6, pp: 1954-1972, **2022**
Doi: 10.1080/00207543.2021.1880656

HYBRID TWO STAGE FLOWSHOP SCHEDULING WITH SECONDARY RESOURCES BASED ON TIME BUCKETS

Abstract

This work studies a two-stage hybrid flowshop problem with secondary resources (workers). The goal is to minimize the average tardiness. The workers are assigned to the workstations by time buckets (work shifts), and the assignment changes during the planning horizon. Two versions of the problem are studied: (i) the case where the average efficiency of the workers determines the time to process jobs; (ii) the case where the efficiency of the slowest worker assigned to a workstation determines the time to process jobs. The problem is NP hard and a set of heuristics are proposed to generate job sequences and worker assignments. Computational experiments are performed on randomly generated test problems. The experiments revealed that the proposed heuristics are able to find a large percentage of the optimal solutions for small sized instances, while on large sized instances the heuristic performance depended on experimental factors.

1. INTRODUCTION

Scheduling jobs and assigning workers to the production equipment is a relevant problem in production environments. The relevance of this problem increases when workers have diverse skills and efficiency levels related to the production equipment and the jobs being manufactured. In many production environments the effective allocation of workers to the tasks at hand is fundamental in meeting the organizations financial and customer service goals.

The motivation of this paper is based on a manufacturer of make to order industrial electrical devices where workers with diverse skill and efficiency levels must be assigned to a set of jobs. In this environment, the process to manufacture/ assemble these devices consist of two stages: the (i) construction/preparation of components and (ii) the assembly of the module(s). These devices (products) are custom-made and belong to product families. Based on customer specifications, work is to be performed in each stage of the process. Orders should be delivered by a due date. There are parallel workstations at each stage. The facility has a set of highly experienced workers that can work in any of the two stages and for any of the product families. The efficiency of each worker is related to the stage in production and to the family of the job processed. The nature of the observed system is one of frequent reconfiguration of their equipment, workspaces, and workforce to meet the ever-changing market needs that come as customer orders. Workforce reconfiguration is an element of many types of production systems (Hashemi-Petroodi et al., 2020).

In the observed environment, workers are assigned to the different workstations per shift. There is a limit of employees working simultaneously in a workstation due to space and tool constraints. For workstations to function properly, they need a minimum number of workers. For example, a workstation can require a minimum of two workers to be able to allow handling of large modules or to administer

certain tests. Figure 1 illustrates the observed environment where the four squares to the left represent the workstations for the first stage of the process, and the three workstations to the right (rectangles) represent the second stage workstations. Eleven workers (A to J), represented by circles, are assigned to the workstations on a daily basis (single shift per day). For Monday's work shift (top section of the figure), three of the first stage workstations are operational with a total of seven workers, and two of the second stage workstations are operational, with four workers. For Tuesday's work shift, the number of operational workstations for stage 1 is reduced by one and the number of workers is also reduced. Furthermore, workstation W1-2 has a different set of workers based on the jobs to be processed that day. For stage 2 the number of operational workstations does not change, however each gets assigned one more worker. Finally, for Wednesday's shift there are no operational workstations in stage 1, and all available workstations in stage 2 are operational. Two of the workstations have four workers, while workstation W2-1 has three, noting that while the number of workers assigned to W2-1 did not change from Tuesday to Wednesday, the set of workers did change. This related to the specific jobs to be processed each day and the ability/efficiency of the workers to these jobs.

Figure 1 here

The effect workers may have on each other when working together in a workstation, may be another relevant element of this environment. When a team is made of workers with significantly different aptitudes/speeds/efficiencies, the overall work tends to be done at the pace of the slowest worker. This may be due to the complexity of the processes and/or dependencies between employees within the workstation. This can be consistent with the concept of bottleneck scheduling (the slowest link controls the speed of the chain!).

Following the observed production environment, this paper addresses a hybrid flowshop (HFS) with secondary resources problem. There are two stages and at each stage there is a predefined number of identical workstations. The set of secondary resources (workers) are assigned to the workstations on a time bucket basis across the planning horizon. In practice, workers were assigned to different workstations every few days as orders were completed and or priorities changed. The process focuses on meeting customer due dates; accordingly, this paper addresses the minimization of average tardiness.

There is an extensive body of work related to the flowshop problem and recent literature reviews have been performed by Neufeld et al. (2016) and Rossit et al. (2018). Many articles published are specific to the hybrid flowshop (HFS) version of the problem since the seminal work of Arthanary and Ramamurthy (1971). Literature reviews that describe the work related to HFS were performed by Linn and Zhang (1999), Ruiz and Vázquez-Rodríguez (2010), Ribas et al. (2010), and Pena-Tibaduiza et al. (2017). The makespan is the most commonly addressed criteria for the HFS and recent articles include Ying and Lin (2018), Fernandez-Viagas et al. (2018) and Öztop et al. (2019). By contrasts, research that addresses problems that considers tardiness criteria (as in this paper) is relatively limited. The article by Guinet and Solomon (1996) address the HFS problem of minimizing the maximum tardiness. The authors propose a set of heuristics and they evaluate their performance against a lower bound on the optimal solution. Lee and Kim (2004), develop dominance properties and lower bound for the total tardiness two stage HFS problem. They consider the case where there is one machine in the first stage, and multiple

machines in the second stage. They develop a branch and bound algorithm to find close to optimal solutions.

Choi et al. (2005) takes on the minimization of total tardiness on the HFS when jobs can return to a previously visited stage. In Khalouli et al. (2010) the goal is to minimize both the tardiness and the earliness of the jobs in a HFS problem. Several heuristics are proposed and evaluated. Yu et al. (2017) develop algorithms for the case of a HFS with batching (and setups). The authors develop a mixed integer programming model for the case where the number of batches is given and then use iterative algorithms to find solutions. The articles by Khare and Agrawal (2019) and Schaller and Valente (2019) consider the combination of earliness and tardiness in the HFS. Khare and Agrawal (2019) develop and evaluate several metaheuristics for the sequence dependent setups case and evaluate their performance versus other well-known metaheuristics. Schaller and Valente (2019) develop heuristics for the case where unforced idle time is allowed as to delay jobs that otherwise would be early. Recently, Yang and Xu (2020) addressed the minimization of delivery and tardiness costs in the distributed permutation flowshop.

Research that considers dual or secondary resources is considerable in the scheduling literature (Wörbelauer et al. 2019), although the work in the flowshop setting is not extensive. Examples include Ruiz-Torres and Centeno (2008), Mehravaran and Logendran (2013), and Figielska (2018). In Ruiz-Torres and Centeno (2008), there are secondary resources that must be allocated across the stages in order to minimize the number of jobs that are late. The paper presents a lower bound process and evaluates several heuristics against this lower bound. Mahravaran and Logendran (2013) consider the assignment of labor to the machines in order to minimize work in process inventory while maximizing the service level. They propose three search algorithms and evaluate their efficiency and effectiveness. Figielska (2018), addresses the two stage flowshop problem as to minimize the maximum completion time where renewable resources are shared among the stages. Heuristics are proposed and computational experiments demonstrate they are of good quality even with strong resource constraints.

The only two articles found in the literature that consider secondary resources in the HFS setting are Figielska (2009) and Figielska (2018), and in both the objective is to minimize the makespan. In Figielska (2009), the problem involves renewable resources in the first stage where multiple machines are available, while the second stage has a single machine and does not require the secondary resource. The problem described in Figielska (2018) involves renewable resources that are shared across the two stages, and there are multiple parallel machines in each stage. In both papers, heuristics based on column generation techniques are proposed and evaluated versus a lower bound.

The research considered in this paper builds on the existing literature related to the HFS problem, in particular, and to reconfigurable production systems in general with many novel elements. This paper considers secondary resources with different, family specific efficiencies, a possibility in many real-world systems with labor intensive operations. As the efficiency of the resource varies, the time to process jobs depends on the specific set of resources assigned to its processing. A unique consideration of this work is modelling the set's efficiency by the worst resource in the group. This consideration has not previously accounted for on the flowshop literature. Furthermore, the assignment of secondary resources is performed by time buckers (worker shifts); an element commonly found in real world production systems and highly relevant in flexible production systems. Managers can reassign workers during the week to different areas to balance work and meet customer needs.

Previous research in the flowshop problem that includes the time buckets characterization was not found in the literature. The next contribution of this research is the modeling of workstation overall efficiency, which determines the jobs process times, based on two relevant cases; aggregate efficiency of the worker set, and slowest efficiency of the worker set. While the second case can represent a significant issue in some industrial settings, it is not considered in previous flowshop literature. This article provides multiple contributions to the HFS body of knowledge of practical relevance as the problem is based on a complex real-world setting.

The remaining of the paper is organized as follows. Section 2 provides the problem description, Section 3 describes a pertinent example, Section 4 presents a set of algorithms aimed to generate feasible schedules, Section 5 describes a set of experiments to evaluate the schedule generation methods, and finally, Section 6 describes the conclusions and provides directions for future research.

2. PROBLEM DESCRIPTION

2.1 Problem Structure

This research considers a two-stage hybrid flowshop with secondary resources where a set of independent jobs must be processed in both stages. The characteristics of the problem to be studied are the following:

- There is a set N of jobs to be scheduled.
- There are two stages and all jobs must be processed in each stage.
- Only one job can be processed at a time in a workstation and no preemption or division of jobs is allowed.
- There is a set U of product families.
- Jobs belong to a product family; let y_j be the family of job j.
- Jobs have a due date; let d_j be the due date of job j.
- Jobs have work content in time units (work volume) for each stage which includes setup and movement time between workstations; let $v_{j,i}$ be the work content of job j in stage i.
- The planning horizon is divided into time buckets of equal duration b.
- There are m_i identical parallel workstations available in stage i.
- There is a set G of secondary resources (workers) to be assigned to the workstations.
- A workstation in stage i must be assigned a minimum number of workers g_i^{min} to be operational.
- A workstation in stage i can have a maximum of g_i^{max} workers.
- Workers are fully cross trained; thus, workers can be assigned to any of the two stages to process any job.
- The assignment of workers to workstations is per time bucket in the planning horizon and let $Z_{s,t}$ represent the set of workers assigned to workstation s during time bucket t.
- Workers can be left unassigned during a time bucket if for example all workstations processing jobs at that time have their maximum possible number of assigned workers. It is assumed that unassigned workers will be performing work in other operations and/or training.

- Workers have different aptitude/speed/efficiency which depend on the stage and the product family; let $e_{w,f,i}$ be efficiency of a worker w to perform a job of family f in stage i.
- Worker efficiencies (the $e_{w,f,i}$) are positive real numbers with a maximum value of 1.
- Workstation efficiency during a time bucket is a function of the workers assigned to it.

The efficiency of a workstation determines the amount of work that is performed during a time bucket t. A workstation's overall efficiency is defined by two different cases: a) *aggregate efficiencies* (AE case) and b) *slowest efficiency* (SE case). The first represents the case where there is no negative interaction between the workers conducting the work in a workstation, while the second represents the case where the slowest worker *limits* the work of all the workers on a workstation. This second case is based on observations made on the industrial case that serves as the basis of this research. Let $\Xi(s)$ be equal to the stage of station s (thus $\Xi(s) = 1$ or 2). When the AE case is assumed, the overall efficiency in workstation s at stage i for family f during time bucket t is defined by $\sum_{g \in Z_{s,t}} e_{g,f,\Xi(s)}$. When the SE case is assumed, the overall efficiency is defined by $|Z_{s,t}| \times min_{g \in Z_{s,t}} e_{g,f,\Xi(s)}$, where $|set|$ is used to indicate the cardinality of the set. The time required to complete work (duration of work) is based on the following relationship:

$$duration\ of\ work = \frac{work\ content}{overall\ efficiency}$$

For example, at the start of a schedule a workstation in stage 1 has been assigned two workers and it will start to process a job from family f. These workers have an efficiency of eight tenths (0.8) and an efficiency of one (1) respectively for family f. The work content for this job is ten (10) hours ($v_{j,1}$ = 10). Consequently, the duration of this job (clock hours) is of five hours and fifty-five minutes [10 / (0.8 +1) = 5.55 hours]. In the discussion of the problem and the example section, the term $\delta hours$ will be used to define clock hours (durations) and $\mu hours$ will be used to define work content hours. Thus, the efficiency variable is a ratio: $\mu hour/\delta hour$.

The schedule (a solution to the problem) consists of two components: (i) the allocation of the workers to workstations on a time bucket basis and (ii) the sequence of jobs in each workstation. Let K_s be the ordered set of jobs (a sequence) in workstation s. The schedule results in completion time for each job in stage 2 where c_j is the completion time (at stage 2) of job j and its tardiness is $max\ [0, c_j - d_j]$. Let Θ represent the average tardiness. Accordingly, the results of this schedule are contemplated as: $\Theta = \sum_{j \in N} max\ [0, c_j - d_j]/n$. The goal and intent of the scheduling process is to minimize the average tardiness.

2.2 Problem Structure Limitations

The problem studied does not consider the utilization of the workers, thus worker's idle time is not measured, although relevant in practice. Workers that are not assigned to a workstation in a complete time bucket or through some segment of a time bucket are assumed to be performing other relevant activities as minor maintenance and training.

2.3 Problem Complexity

As mentioned in Pinedo (2016), the flowshop problem with two machines for the total completion time objective ($F2||\sum c_j$) is NP-Hard, therefore, based on complexity hierarchy, the two machine flowshop problem with total tardiness objective is NP Hard. Similarly, based on complexity hierarchy, the hybrid flowshop problem is more complex than the "base" flowshop problem. Hence, it is concluded that the problem addressed in this research is NP-Hard. We also note that our problem characterization includes secondary resources and time buckets, which increase the problem's complexity. This complexity serves as a way to characterize many elements of a real-world setting.

2.4 Summary of Notation

Indexes

i	stages
w	workers
s	workstations
f	families
j	jobs
t	time buckets

Parameters

m_i	Number of available parallel workstations on stage i
X_i	Set of available parallel workstations in stage i
G	Overall set of workers
g_i^{max}	Maximum number of workers that can be assigned to a workstation on stage i
g_i^{min}	Minimum number of workers that can be assigned to a workstation on stage i
b	Duration of a time bucket in hours
N	Set of all jobs to be scheduled
U	Set of all product families
d_j	Due date for job j
y_j	Product family of job j
$v_{j,i}$	Work content for job j on stage i in hours
$e_{w,f,i}$	Efficiency of the worker w while performs a job of family f on stage i ($\in(0,1]$)
$\Xi(s)$	Stage of station s (thus $\Xi(s) = 1$ or 2)
c_j	Completion time of job j on stage 2
Θ_Ψ	Average tardiness of a schedule Ψ

Decision Variables

$Z_{s,t}$	Set of workers assigned to workstation s during time bucket t
K_s	Ordered set (sequence) of jobs on workstation s

3. ILLUSTRATION CASE

An example is provided to illustrate the problem. There are 6 jobs to be scheduled and the jobs belong to one of two families. There are 3 parallel workstations available at each stage, 7 workers, and the duration of each bucket is 8 hours ($m_1 = m_2 = 3$, $G = \{\text{w1}, \text{w2}, \text{w3}, \text{w4}, \text{w5}, \text{w6}, \text{w7}\}, b = 8\ \delta hours$). Let X_1 = {s1-1, s1-2, s1-3} and X_2 = {s2-1, s2-2, s2-3}. The minimum and maximum workers per workstation in stage 1 is 1 and 4 respectively, while for stage 2 is 1 and 3 ($g_1^{min} = 1, g_1^{max} = 4, g_2^{min} = 1, g_2^{max} = 3$). The information regarding the jobs and the workers is presented in Tables 1 and 2 respectively.

Tables 1 and 2 here

3.1 Example Schedule and Worker Assignments

The production planner has developed job sequences and worker assignments based on personal logic and experience. The planner has determined that for this planning horizon two workstations are to be operational for each of the two stages. Furthermore, the planner has decided on the following job sequences in the workstations: $K_{s1-1} = j1 - j3 - j5$; $K_{s1-2} = j2 - j4 - j6$; $K_{s2-1} = j1 - j3 - j6$; and $K_{s2-2} = j2 - j4 - j5$.

This set of sequences is called S1. The planner has decided in the following worker assignments: workers w1 and w7 will start at workstation s1-1 and change to workstation s2-2 at $t = 6$; workers w2 and w5 are assigned to workstation s1-2 for $t = 1 \ldots 4$; then worker w2 is assigned to s2-1 for all future time buckets, while w5 is left with the same assignment; workers w4 and w6 are assigned to workstation 1-2 for $t = 1 \ldots 2$; and then to station s2-1 for all future time buckets; and worker w3 is assigned to workstation s1-1 for $t = 1$ and then assigned to workstation s2-2 for all future time buckets.

Figure 2 presents (i) the workstation sequences and (ii) the worker assignments previously described. The shaded areas represent stage two (2) workstations. Part (iii) of Figure 2 presents the schedule that "combines" the job sequences (S1) with the worker assignments (A1) and assumes the AE workstation overall efficiency case. Table 3 provides the start, the duration, and the end times of each job in each stage for this schedule. All values in the table are in δhours.

Figure 2 here

Table 3 here

The process and calculations to determine the start and end times for the first two jobs assigned to workstation s1-2 are described next.

> Job j2 starts on workstation s1-2 at clock time 0 in time bucket 1 ($t = 1$). Job j2 has a work content of 26 μhours and belongs to family 2 (see Table 1: $v_{j,1}$ = 26 $\mu hours$, f_j = 2). During $t = 1$, workers w2, w4, w5 and w6 are assigned to workstation s1-2.

worker	$e_{g,f2,1}$
w2	0.65
w4	0.9
w5	0.8
w6	1.0
Sum	3.35

The sum of the efficiencies for this set of workers for family 2 is 3.35, therefore the overall efficiency of the workstation is 3.35 $\mu hour\ /\ \delta hour$. The amount of work this set of workers performs on job j2 during $t\ =\ 1$ is 26.8 μhours ($3.35\ \times b$). As the work content of j2 (26) is less than or equal to the work that can be completed by the workers (26.8), j2 will be completed during $t\ =\ 1$. The time to complete the work (duration) is determined by:

$$= \frac{work\ content}{overall\ efficiency} = \frac{26\ \mu\text{hours}}{3.35\ \mu\text{hour}\ /\ \delta\text{hour}} = 7.761\ \delta\text{hours}$$

The end clock time for j2 in stage 1 is 7.761 ($c_{j2,1} = 7.761$). It is important to note that this job will not immediately start on stage 2 as there are no workers in any of the workstation of that stage at $t\ =\ 1$.

Job j4 ($v_{j,1}$ = 15 $\mu hours$, f_j = 2) starts immediately after j2 is finished in s1-2 and there are 0.239 δhours still available in this time bucket (8 – 7.761). As job j4 is from family 2, the overall efficiency is 3.35 $\mu hour\ /\ \delta hour$. Therefore, 0.8 $\mu hours$ of work content are completed for j4 ($3.35\ \times 0.239$) during $t\ =\ 1$. Given j4 has 15 $\mu hours$ of work content, j4 is not finished during t = 1, and there are 14.2 $\mu hours$ still to be completed after the end of this bucket.

At $t\ =\ 2$, the same set of workers is assigned to s2-1, thus the overall efficiency does not change. The amount of work content that can be completed at $t\ =\ 2$ is 26.8 μhours and given 26.8 $\mu hours \geq$ 14.2 $\mu hours$ (the work left from $t\ =\ 1$), j4 will be completed during $t\ =\ 2$.

The time to complete the work (duration) at $t\ =\ 2$ is determined by:

$$= \frac{work\ content}{overall\ efficiency} = \frac{14.2\ \mu\text{hours}}{3.35\ \mu\text{hour}\ /\ \delta\text{hour}} = 4.239\ \delta\text{hours}$$

The end time for j4 in stage 1 is 12.239 ($c_{j4,1} = 12.239$). The sum of 8 from ($t\ =\ 1$) and 4.239 (duration during $t\ =\ 2$). Therefore j4 is completed in this station during time bucket 2.

Appendix 1 provides additional examples of the calculations of the start and end time of the schedule in station 1-2. Table 4 provides the job-related measures. The average tardiness is 3.632. Finally,

it is noted that this schedule is not an optimal solution to this problem and none of the algorithms under consideration are guaranteed to find the optimal solution.

Table 4 here

3.2 Schedule under SE case assumption

This section describes the resulting schedule under the assumption of the SE case. Therefore, it is now assumed that the slowest worker dictates the amount of work that is performed in a workstation during a time bucket. Figure 3 presents two schedules. The top schedule is the same as that shown in Figure 2 (resulting from combining S1 and A1 under the AE case) while the bottom schedule is the result of combining S1 and A1 under the SE case. The schedule under the AE is presented again in Figure 3 to provide a clear examination of the schedule differences due to the two assumption regarding the overall efficiencies of the workstations.

Figure 3 here

Table 5 presents the start, duration, and end times for all the jobs in both stages under the SE case. The majority of the durations are, as expected, longer under the SE case. The few exceptions relate to jobs that are processed in different time buckets with different worker sets.

Table 5 here

To demonstrate the differences between the AE and SE assumptions, the determination of the completion time of job j2 in workstation s1-2 is described next.

At $t = 1$ job j2 ($v_{j,1}$ = 26 $\mu hours$, f_j = 2) starts on workstation s1-2 at clock time 0. During $t = 1$, workers w2, w4, w5, w6 are assigned to workstation s1-2.

worker	$e_{g,f2,1}$
w2	0.65
w4	0.9
w5	0.8
w6	1.0
Minimum	0.65

The minimum efficiency for this set of workers for family 1 is 0.65 and given there are 4 workers, the overall efficiency of the workstation is 2.6 $\mu hour$ / $\delta hour$ (0.65×4). Note: in this situation it would make sense to remove w2 from the workstation altogether; it is clearly not a good set of workers to put together. The amount of work this set of workers performs on job j2 during $t = 1$ is 20.8 μhours ($2.6 \times b$). Given j2 has 26 $\mu hours$ of work content, j2 is not finished during $t = 1$, and there are 5.2 $\mu hours$ still to be completed after the end of this bucket.

At $t = 2$, the worker assignment is not changed and based on the previous calculations j2 can be completed during $t = 2$. The time to complete the work (duration) at $t = 2$ is determined by $(5.2\ \mu\text{hours})/(2.6\ \mu hour\ /\ \delta hour) = 2\delta\text{hours}$. The end time for j2 in stage 1 is at clock time 10. By comparison, under the AE case, j2 was completed at clock time 7.76, 2.24 hours earlier.

Table 6 provides the completion time and tardiness for each job. The average tardiness is 5.888. As in the schedule presented for the AE case, this schedule is not an optimal solution to this problem.

Table 6 here

4. SOLUTION METHODS

The problem as described requires decisions regarding the number of workstations to have operational for each stage, the sequence of orders to the workstations, and the assignment of the workers. The assignment of workers to the workstations is performed in a per time bucket basis (for example a shift). This section presents a set of sequencing and resource allocation rules aimed at generating feasible and effective solutions. It is noted that the authors know of no previously proposed solution methods for this particular version of the hybrid flowshop problem that can be used as a comparison point.

The section is divided into two subsections, the first detailing how initial schedules are developed, and the second describing approaches to improving the schedules through worker reallocation.

4.1 Initial Schedule Development

The initial schedule building process is divided into three phases. The first phase involves determining the number of workers that will be initially allocated to each stage and determining the number of workstations that will be operational. The second phase performs an initial assignment of workers to operational workstations. The third phase of the initial schedule involves assigning workers to each workstation and developing job sequences in each operational workstation.

Phase 1. Initial number operational workstations per stage.

Step 1. Determine the target number of workers per stage based on the proportion of total workload volume and modify such that at least one workstation is operational for each stage, and so all workers can be allocated to a workstation. Let $V1$ and $V2$ be total workload volume for stages 1 and 2 respectively and ρ_i be the number of workers initially assigned to stage *i*.

1.1. Calculate the total workload per stage: $V1 = \sum_{j \in N}[v_{j,1}]$, $V2 = \sum_{j \in N}[v_{j,2}]$.

1.2 Calculate the proportional number of workers per stage: $\rho_1 = \lceil(|G| \times V1)/(V1 + V2)\rceil$ and $\rho_2 = \lfloor(|G| \times V2)/(V1 + V2)\rfloor$.

1.3 Calculate the number of workers needed to have one operational workstation per stage: $\alpha_i = max\ [0, g_i^{min} - \rho_i]$ for $i = 1, 2$.

1.4 Let $\rho_1 = \rho_1 + \alpha_1 - \alpha_2$ and $\rho_2 = \rho_2 + \alpha_2 - \alpha_1$.

1.5 Calculate the *extra* workers based on the maximum possible number of workstations per stage: $\varepsilon_i = max\ [0, \rho_i - x_i \times g_i^{max}]$ for $i = 1, 2$.

1.6. Let $\rho_1 = \rho_1 + \varepsilon_2 - \varepsilon_1$; and $\rho_2 = \rho_2 + \varepsilon_1 - \varepsilon_2$.

Determine the number of operational workstations for each stage, where γ_i is the number of operational parallel workstations in stage *i*. This is done by one of three rules: the first determines the largest number possible for both stages; the second rule determines the smallest number possible for both stages, the third determines the largest and smallest number of workstations for stages 1 and 2 respectively. Note that a rule where the smallest and largest number of workstations for stages 1 and 2 respectively was examined, but pilot experiments demonstrated it generated poor results thus not considered further.

(Rule 1).
1.7. Let $\gamma_i = min\,[x_i, \lfloor \rho_i / g_i^{min} \rfloor]$ for $i = 1, 2$.
(Rule 2).
1.7. Let $\gamma_i = \lceil \rho_i / g_i^{max} \rceil$ for $i = 1, 2$.
(Rule 3).
1.7. Let $\gamma_1 = min\,[x_1, \lfloor \rho_1 / g_1^{min} \rfloor]$ and $\gamma_2 = min\,[x_2, \lceil \rho_2 / g_2^{max} \rceil]$

Phase 2. Initial worker assignment
Step 2. Initialize the sets for workers and workstations. Calculate the average efficiency for each worker and the control ratio, a variable used to determine the stage that will be assigned the next worker. Let Γ_i be the set of operational parallel workstations in stage *i*.

2.1 Let $G' = G$ and $Z_{s,1} = \emptyset\ \forall s \in X_i, i = 1, 2$.
2.2 Select any γ_i worstations from X_i and assign to set Γ_i for $i = 1, 2$.
2.3 Let $e_{w,i}^{ave} = [\sum_{f \in U} e_{w,f,i}]\,/f\ \forall w \in W, \forall i = 1, 2$.
2.4 Let $\sigma_{control} = \rho_1\,/(\rho_1 + \rho_2)$.

Step 3. Select the stage to assign the next worker based on smallest absolute difference to the control ratio. Assign the selected worker to the workstation on that stage with the minimum number of workers. End this phase when all workers have been assigned. Note that $abs[x]$ is used to indicate absolute value.

3.1 Let $\sigma_1 = (\rho_1 - 1)\,/(\rho_1 + \rho_2 - 1), \sigma_2 = \rho_1/(\rho_1 + \rho_2 - 1)$.
3.2 If $abs[\sigma_1 - \sigma_{control}] > abs[\sigma_2 - \sigma_{control}]$ and $\rho'_1 > 0$ then $i' = 1$, else $i' = 2$.
3.3 Let $w' = \{w \mid w \in G', max\,[e_{w,i'}^{ave}]\}$, $s' = \{s \mid s \in \Gamma_{i'}, min\,[|Z_{s,1}|]\}$ and $\rho_{i'} = \rho_{i'} - 1$.
3.4 Let $Z_{s',1} = Z_{s',1} \cup w'$ and $G' = G' - w'$.
3.5 If $|G'| > 1$ then return to 3.1.
3.6 Let $w' = \{w \mid w \in G'\}$.
3.7 If $\rho_1 = 1$ then $s' = \{s \mid s \in \Gamma_{1'}, min\,[|Z_{s,1}|]\}$ else $s' = \{s \mid s \in \Gamma_{2'}, min\,[|Z_{s,1}|]\}$
3.8 Let $Z_{s',1} = Z_{s',1} \cup w'$
3.6 Let $Z_{s,t} = Z_{s,1}\ \forall s \in \Gamma_i\,, t = 2 \ldots \infty, i = 1, 2$.

Phase 3. Job assignment to workstations

Step 4. Initialize the set for unassigned jobs and the ordered set of jobs for the workstations in stage 1. Assign jobs from the set of unassigned jobs by minimum due date (ties solved arbitrarily) to the workstations, selecting the workstation where the job will be completed first. The function $CT(s)$ determines the clock time of the ordered set of jobs assigned to workstation s.

4.1 Let $N' = N$ and $K_s = \emptyset \; \forall s \in \Gamma_1$.
4.2 Let $j' = \{j \mid j \in N', min\left[d_j\right]\}$.
4.3 Let $N' = N' - j'$.
4.4 Add j' to the next position of $K_s \forall s \in \Gamma_1$.
4.5 Let $s' = \{s \mid s \in \Gamma_1, min\,[CT(s)]\}$.
4.6 Let $K_s = K_s - j' \forall s \in \Gamma_1$ and add j' to the end of $K_{s'}$.
4.7 If $N' \neq \emptyset$ return to 4.2.

Step 5. Initialize the set for unassigned jobs and the ordered set of jobs for the workstations in stage 2. Assign jobs from the set of unassigned jobs by completion time in stage 1 (ties solved arbitrarily), selecting the workstation based on one of two rules. The first rule selects the workstation that is available first, the second rule selects the workstation where the job will be completed first.

5.1 Let $N' = N$. Let $K_s = \emptyset \; \forall s \in \Gamma_2$.
5.2 Let $j' = \{j \mid j \in N', min\left[c_{j,1}\right]\}$.
5.3 Let $N' = N' - j'$.

(Rule 1)
5.4 Let $s' = \{s \mid s \in \Gamma_2, min\,[CT(s)]\}$. Add j' to the next position of s'.
5.5 If $N' \neq \emptyset$ return to 5.2.

(Rule 2)
5.4 Add j' to the next position of $K_s \forall s \in \Gamma_2$.
5.5 Let $s' = \{s \mid s \in \Gamma_2, min\,[CT(s)]\}$.
5.6 Let $K_s = K_s - j' \forall s \in \Gamma_1$ and add j' to the end of $K_{s'}$.
5.7 If $N' \neq \emptyset$ return to 5.2.

4.2 Improvement Strategies

Two worker reassignment procedures based on neighborhood search methods are presented in this section. They are based on two algorithms described separately, the first searches using single job reinsertions and the second uses pairwise exchanges. Let Ψ represent the current schedule (sequence of jobs assigned to each workstation and worker assignment to each workstation per time bucket). This schedule has an average tardiness of Θ_Ψ and all jobs are completed by time bucket $t_{max}(\Psi)$.

Algorithm 1. Insert(τ)

This algorithm determines the workstation that has assigned the jobs with the highest total tardiness that are finished after time bucket τ. The process considers adding workers to that workstation and the process continues until it cannot find improvements for insertions based on time bucket τ. Let g_s^{*max} be the maximum number of workers allowed for workstation s, where $g_s^{*max} = g_1^{max}\ \forall s \in \Gamma_1$ and $g_s^{*max} = g_2^{max}\ \forall s \in \Gamma_2$. Let g_s^{*min} be the minimum number of workers allowed for workstation s, where $g_s^{*min} = g_1^{min}\ \forall s \in \Gamma_1$ and $g_s^{*min} = g_2^{min}\ \forall s \in \Gamma_2$.

Inputs τ, Ψ

A1.1 Let $\Omega = \Gamma_1 \cup \Gamma_2$.

A1.2 Let $\varphi_s = \sum_{j \in K_s} \theta_j | c_{j,1} \geq m\ \forall s \in \Gamma_1$ and $\varphi_s = \sum_{j \in K_s} \theta_j | c_{j,2} \geq b\ \times\ \tau.\ \forall s \in \Gamma_2$.

A1.3 Let $s' = \{s\ |s \in\ \Omega, max\ [\varphi_s], |Z_{s,\tau}| < g_s^{*max}\}$.

A1.4 If $\varphi_{s\prime} = 0$ or $s' =\ \emptyset$ then End else $\Omega = \Omega - s'$.

A1.5 Let $G' =\ G - Z_{s\prime,\tau}$.

A1.6 Let $G' =\ \{\ G' - \cup_{s \in \Omega} Z_{s,\tau}\ | |Z_{s,\tau}| = g_s^{*min}\}$.

A1.7 Generate the set Ξ of $|G'|$ temporary schedules by adding each worker in G' to workstation s' during time bucket τ.

A1.8 Let $\theta' = \{\theta\ |\theta \in\ \Xi, min\ [\Theta_\theta]\}$.

A1.9 If $\Theta_\Psi > \Theta_\theta$, then let $\Psi =\ \theta'$ and go to Step A1.1.

A1.10 If $\Omega = \emptyset$ then End, else go to step A1.3.

Algorithm 2. Exchange(τ)

This algorithm determines the workstation that has assigned the jobs with the highest total tardiness that are finished after time bucket τ. The process considers exchanges between workers in that workstation and all others until it cannot find improvements for exchanges based on time bucket τ.

Inputs τ, Ψ

A2.1 Let $\Omega = \Gamma_1 \cup \Gamma_2$.

A2.2 Let $\varphi_s = \sum_{j \in K_s} \theta_j | c_{j,1} \geq b\ \times\ \tau\ \forall s \in \Gamma_1$ and $\varphi_s = \sum_{j \in K_s} \theta_j | c_{j,2} \geq b\ \times\ \tau\ \forall s \in \Gamma_2$.

A2.3 Let $s' = \{s\ |s \in\ \Omega, max\ [\varphi_s]\}$.

A2.4 If $\varphi_{s\prime} = 0$ then End else $\Omega = \Omega - s'$.

A2.5 Let $G' =\ G - Z_{s\prime,\tau}$.

A2.6 Generate the set Ξ of $|Z_{s',\tau}|\ \times |G'|$ temporary schedules by exchanging each worker in $Z_{s\prime,\tau}$ with each worker in G' during time bucket τ.

A2.7 Let $\theta' = \{\theta\ |\theta \in\ \Xi, min\ [\Theta_\theta]\}$.

A2.9 If $\Theta_\Psi > \Theta_\theta$, then let $\Psi =\ \theta'$ and go to Step A2.1.

A2.9 If $\Omega = \emptyset$ then End, else go to step A2.3.

Phase 4. Search for improved schedules

Step 6. Given an input schedule Ψ generate new schedules by changing the worker assignments. This is done by one of two rules: the first rule implements Insert(τ) one bucket at a time for all time buckets, and then implements Exchange(τ) one bucket at a time for all time buckets; the second rule implements Insert(τ) and then implements Exchange(τ) one bucket at a time for all time buckets

Input: Ψ

Step 6. (Rule 1)

6.1 For τ = 1 to $t_{max}(\Psi)$
  Perform Insert(τ).
Next.

6.2 For τ = 1 to $t_{max}(\Psi)$
  Perform Exchange(τ).
Next.

Step 6. (Rule 2)

6.1 For τ = 1 to $t_{max}(\Psi)$.
  Perform Insert(τ).
  Perform Exchange(τ).
Next

It is noted that during the execution of both Insert(τ) or Exchange(τ), the current schedule Ψ may change, and therefore the value of $t_{max}(\Psi)$; the end point of the For-Next loop. Given the current formulation, there is no possibility of a situation in the execution of the algorithms where a new schedule has a $t_{max}(\Psi)$ that is smaller than the current bucket τ, therefore no backtracking is necessary.

4.3 Algorithms

The solution method consists of the four phases described in sections 4.1 and 4.2 and summarized by the diagram in Figure 4. There are a total 6 steps where Step 1 has three possible rules to determine the number of operational workstations per stage, Step 5 has two possible rules to assign jobs to workstations in stage 2, and Step 6 has two possible rules to search for improved worker assignments. This results in 12 possible combinations of rules, and the description of each combination is provided in Table 7.

Figure 4 here

Table 7 here

4.4 Examples

The implementation of three of the algorithms for the example described in section 3 are presented in this subsection. Figure 5 presents the schedule resulting from G1, G6, and G12 under the SE case. The schedules are significantly different (from each other) as they are based on a different number of operational workstations. Job sequences are also different across the three schedules except in the case of

G1 and G12 where the sequences are similar for stage 1. However, the completion times at each stage for each of the jobs is different based on the particular worker assignments. Tables 8 to 10 present the worker assignments for G1, G6, and G12 respectively, noting that "--" indicates the worker was fully idle during that time bucket. The assignments are significantly different across the three schedules and the workers allocated across multiple workstations during the 8 time periods. Only G6 has a case where a worker stayed in the same workstation for the duration of the schedule (worker w7). The average tardiness for G1, G6, and G12 are 4.1, 3.16, and 5.08 respectively, where G6 generates the best solution out of the 12 algorithms.

Figure 5 here

Tables 8-10 here

5. COMPUTATIONAL EXPERIMENTS

This section describes the results of two experiment sets designed to evaluate the performance of the proposed rules. Similar to the approach used in Yu et al. (2017), the completed analysis evaluates algorithm performance for small instances versus an optimal solution, while for large instances it evaluates the algorithm's relative performance. The algorithms were coded in Visual Basic for Applications running in Excel. The experiments were conducted on a personal computer with a 2.9GHz processor and 12GB of RAM memory using the Windows 10 operating system. All instances are based on randomly generated numbers. As in Wang et al. (2016) and Neufeld et al. (2020), this problem is novel (includes characteristics not previously addressed in the literature) thus there are no existing methods that can be used as benchmark/comparison points, and the experiments focus on characterizing relative performance among the proposed versions of the solution approach.

5.1 Performance versus the optimal

This experimental framework is designed to test the ability of the proposed algorithms to find an optimal solution. Given the complexity of the problem, these experimental instances were created by inverse construction, starting with an optimal solution with a known tardiness of 0 and based on randomly generated efficiencies and other characteristics of the instances. The due dates were set equal to each of the job's completion times on stage 2 and the instance's data (efficiencies, work volumes, due dates) is then an input to the algorithms. Since the optimal tardiness value for the instance is known (it is 0), this approach is able to determine if the algorithms are able to generate a schedule with the optimal tardiness measure. For these experiments, the aggregate efficiency case (AE) is assumed, the duration of the time bucket is 8 hours ($ie.b = 8$), the minimum number of workers per station is always 1, ($ie.w_1^{min} = w_2^{min} = 1$), the efficiency of a worker g for jobs of family h for a stage i is defined by a randomly generated value from a uniform distribution; $e_{g,h,i} = U(0.5,1)$, and there are 2 families ($ie.f = 2$).

Three experimental factors are considered at two levels to examine how the characteristics of the instances influence the algorithms performance. These are presented in Table 11. The first two factors refer to the complexity defined by problem size. The first factor (F1), relates to the number of jobs ($n =$

$|N|$). The second factor (F2), relates to the number of workers, the maximum number of workers per workstation, and the number of parallel stations per stage (Level 1 being a small shop and Level 2, a medium shop). Factor F3, relates to the work-content of the jobs and their possible complexity, modeled by the divergence of the processing times in the resulting schedule. This is modeled by the range of the processing times in the stations. The values of the process time in the stations (in hours) were generated using a random uniform distribution, where level 1 indicates that the process times are relatively homogeneous, and in level 2, where they are relatively heterogeneous.

Table 11 here

The number of times an optimum solution (with tardiness of 0) is found by all the algorithms and by at least one of them is presented in Table 12, for each experimental point. A summary of the results by experimental variable is presented in Table 13. All the algorithms found the optimal solution in 13 out of the 40 instances, while in 30 out of the 40 instances at least one algorithm found the optimal solution. It is noted that the tardiness values are less than an hour for the instances where the optimal solution is not found by any of the algorithms. It is argued that as a set, finding the optimal solution in 75% of the cases in combination with low tardiness values when the optimal is not found indicate they perform well in terms of generating close to optimal solutions under the scope of this set of experiments. Reviewing the results from Table 13, and focusing on the number of times at least one of the algorithms finds an optimal solution, it is noted that fewer optimal solutions were found at Level 1 of F2 (*smaller* shop) and Level 2 of F3 (heterogeneous process times).

Tables 12 and 13 here

5.2 Relative performance

The second experimental framework is designed to evaluate the relative performance of the algorithms for larger problems and where the optimal solution is not known. In these experiments, there are 5 parallel workstations for each stage ($x_1 = x_2 = 5$). The minimum and maximum number of workers that can be assigned per workstation is 1 and 5, respectively, and this applies to both stages ($ie. w_1^{min} = w_2^{min} = 1,\ w_1^{max} = w_2^{max} = 5$). There are f families, where the value is set to 4. Pilot experiments with values of $f = 2$, 3, and 6, lead to no changes in average tardiness performance for the algorithms. The efficiency of a worker g for jobs of family h for a stage i is defined by a randomly generated value from a uniform distribution: $e_{g,h,i} = U(0.5,1)$. The duration of the time bucket is 8 hours ($ie. b = 8$).

Three experimental factors are considered: two of them at two levels and one at three levels. These are presented in Table 15. The first factor (F1) relates to the instance complexity (problem size) accounting for the number of jobs and workers (Level 1 representing a medium shop, and Level 2 a large shop). The second factor (F2) considers the effect of the balance in the work-content of the jobs; balanced across the stages (level 1), unbalanced where there is less work in the first stage of the flowshop (Level 2), and unbalanced where there is less work in the second stage of the flowshop (Level 3). The third experimental factor (F3) considers the effect of due date tightness on relative performance modeled by a tightness ratio; low tightness (Level 1) and high tightness (Level 2).

The implementation of these factors is completed as follows. For each job, the volume of work content per stage is defined by a randomly generated value from a uniform distribution $v_{j,1} = U(vol_1, 2 \times vol_1)$ and $v_{j,2} = U(vol_2, 2 \times vol_2)$, where vol_2 and vol_2 are defined per F2 (see Table 15). The total work content of the shop is $vol_{total} = \sum_{j \in N}[v_{j,1} + v_{j,2}]$ and the due date of each job is defined by a randomly generated value from a uniform distribution $d_j = vol_1 + vol_2 + U(0, vol_{total}/TR)$, where the value of *TR* (tightness ratio) is defined by experimental factor F3 (see Table 14). The tightness ratio values were selected based on pilot experiments as to have approximately 15% and 40% jobs late respectively under the first level of both F1 and F2 and based on the AE case assumption. There is a total of 12 experimental combinations ($2 \times 3 \times 2$), and for each combination 20 problem instances were generated.

Table 14 here

5.2.1 Experimental Results in the AE case assumption

The results when considering the assumption of aggregate efficiency (AE) are presented in Table 15. The table presents per experimental level the percentage of instances where each algorithm generated the best solution, and in parenthesis, the average tardiness. This discussion focuses on characterizing relative performance by the percentage of times an algorithm finds the best solution. The cells in grey indicate the best performing algorithm for each level of each experimental factor. The last column of the table provides the overall results. Algorithm G6 is the best overall performer and in general, it is observed that algorithms G5 to G8 are the best performers across the experiments, all at least finding 20% of the best solutions. These four algorithms share as a common element being based on Rule 2 for Step 1, which sets the number of operational workstations to the minimum possible for both stages. Note that the percentages in each column add to more than 100% given more than one algorithm can determine the best solution for an instance (ties). It also noted that algorithms G1 to G4 performed poorly, finding none of the best solutions in these experiments. The common element in algorithms G1-G4 is that they are all based on Rule 1 for Step 1 which sets the number of operational workstations to the maximum possible in both stages. Clearly, this approach does not work well in the AE case.

Table 15 here

The experimental factors play a role in algorithm relative performance. When considering F1 (shop size), at Level 1 (the medium shop) algorithm G6 is the best performer finding 33% of the best solutions, while G10 is the best performer at L2 (the large shop), finding 23% of the best solutions. This is the only case where an algorithm outside the G5-G8 group dominates. Algorithm G10 is based on Rule 3 for Step 1, which sets the largest number of operational workstations in stage one and the minimum in the second stage. When considering F2 (workload balance), at Level 1 (balanced workload) algorithm G8 dominates, at Level 2 (lower workload in stage 1) algorithm G5 dominates, and at Level 3 (lower workload in stage 2) algorithm G6 dominates. When considering F3 (due date tightness), algorithm G6 dominates al Level 1 (low tightness) and G5 at Level 2 (high tightness). In general, the results demonstrate that algorithm performance depends on shop characteristics where no single approach

outperforms the rest. However, the difference in relative performance when looking at the tardiness measure can be small for the top 2-3 algorithms.

5.2.2 Experimental Results in the SE case assumption

Table 16 presents the results when the slowest efficiency (SE) case assumption is considered, and this table presents the same information as Table 15. Algorithm G2 is the best overall performer generating 31% of the best solutions, followed by G4, which generated 21% of the best solutions. When comparing these results to those of the AE case, it is noted that no single or group of algorithms was completely outperformed. In other words, all 12 algorithms were able to generate some of the best solutions (no algorithm had 0% in the overall column).

Table 16 here

The algorithms' relative performance is also related to the experimental factors in the SE case. At Level 1 of F1 (the medium shop) algorithm G6 is, similarly to in the AE case, the best performer finding 26% of the best solutions. Generating 46% of the best solutions, algorithm G2 is the best performer at L2 (the large shop). This is the only factor where one of the algorithms dominates in an experimental level for both the AE and SE case (Level 1 of F1). Similarly to the AE case experiments, three different rules dominate across the three levels of F2, but none of the algorithms is the same as those that dominated previously. Furthermore, those that dominated in the AE case performed poorly in the SE case. At Level 1 (balanced workload) algorithm G4 dominates, at Level 2 (lower workload in stage 1) algorithm G10 dominates, and at Level 3 (lower workload in stage 2) algorithm G2 dominates. The experiments related to the Level 3 of F2 are noteworthy as they represent the only condition where an algorithm generates more than 50% of the best solutions. The results associated to F2 are also notable as they include the only case where the best solution relates to having the maximum number of workstations operational in stage 1 and the minimum in stage 2. The result is logical as it relates to the experiment level where the higher workload is in stage 2. The results for F3 represent the only condition where one algorithm dominates across all the experimental levels, moreover by notable differences (the next best performing algorithm is several percentage points below).

5.3 Discussion and managerial implications

The analysis of the results across the two sets of experiments served two objectives: (1) to estimate the algorithms' ability to generate optimal solutions and (2) to determine their relative performance, both with the macro objective of identifying the value of implementing these algorithm as practical industrial solutions. The results demonstrated that the algorithms, as a set, can generate a relatively large percentage of optimal solutions and that the error is small for those where it was not found. However, this conclusion is only certain within the small/constrained experimental framework described in section 5.1. The results also demonstrated that none of the algorithms is the "best" approach, and instead, multiple algorithms should be used in the industrial implementation as their performance is dependent on particular shop characteristics.

Given computational times for algorithms could be significant, the analysis must consider both, which algorithms typical perform well and which ones are poor performers in order to determine which

algorithms to run (and those not to run) for problem instances as to maintain reasonable running times. In the case of the AE assumption, algorithms G5 to G8, which are based on having the minimum number of workstations per stage, performed very well, while algorithms G1 to G4, which are based on having the maximum number of workstations per stage, were highly ineffective. This clearly indicates that having the right number of workstations operational is very relevant in the AE case, and having the workers *spread out* is not an effective strategy.

In the case of the SE assumption, algorithms G2 and G4 were highly effective, while algorithms G1, G3, G5, G7, and G11 were ineffective. A common element in G2 and G4 is that both are based on the maximum number of workstations per stage. Therefore, contrary to the AE case, having the workers spread out is an effective strategy in the SE case. The five ineffective algorithms share as a common component Step 6, where the reassignment is first performed for all buckets and then the exchange of workers is performed for all buckets.

Given the relevance of the AE versus SE assumption in the efficacy of the algorithms, a key managerial decision is to determine which of the two assumptions is a more valid representation of their operations. Management would also need to evaluate the reconfiguration options for equipment/workstations to control the maximum and minimum level of worker reallocations that are possible. This also has an effect on performance. Furthermore, management must consider how due dates for jobs are assigned given their implications to on-time delivery and the level of worker reassignments it may require.

6. CONCLUSIONS AND FUTURE WORK

The flowshop problem has been the subject of extensive research in many variants which consider diverse processing and shop characteristics (Neufeld et al., 2016, Rossit et al., 2018). This research proposes a new variant based on an industrial setting that includes flexibility in terms of which workstations are operational per stage, the allocation of the workers to the workstations, and the assignment of the workers by time buckets to the schedule. All of these characteristics make for a complex problem that requires solution approaches with multiple stages. This work described in detail multiple algorithms that address the characteristics of the proposed flowshop problem, which includes determining the number of workstations to have available and how to reassign workers across the workstations on a time bucket basis.

This research has several contributions to the body of knowledge in scheduling. First, it considers the effect of worker efficiency in the duration of a schedule based on time buckets where the workers can be allocated to a different workstation each bucket. No research that considers such characteristic is known to the authors, even when this is a real-world characteristic of some systems. There are many real-world applications where workforce reconfiguration is/could be performed in a per shift basis and where the approaches described in this paper could be used. This research is also relevant because it models the case where the slowest (or bottleneck) worker is the controlling factor on the time to process a job. It is proposed that in a variety of real-world settings with labor intensive team activities, the described "slower" worker effect may be relevant and for the most part has been ignored by the scheduling literature.

There are many promising directions of future work that builds on the concepts of determining the operational workstations, assigning flexible workers to workstations per time bucket, and worker

efficiency considering the slowest worker of a group. For example, research into generating robust schedules based on different levels of efficiency, similar to the scenario model in Wu et al. (2020). Furthermore, the production environment that serves as a basis for this work is also evolving to include some automation, thus it would be interesting to consider the effect of additional resources where efficiency does not change. Another related direction would be to model only one of the stages as a parallel machine setting and focus on the theoretical setting that could determine the optimal number of workstations to have operational, based on some limited characterization of the problem. Future research work on the applied sense would focus on the application of the proposed algorithms to the described environment.

ACKNOWLEDGEMENTS

The authors wish to thank the associate editor and anonymous reviewers for their insightful comments. This research was partially carried out with the financial support of the Spanish Ministry of Economy, Industry and Competitiveness, and the European Regional Development Fund (ERDF), grant DPI2017-85343-P.

REFERENCES

Arthanary, T. S. .& Ramamurthy, K.G. (1971). An extension of two machine sequencing problem. *Opsearch*, *8*, 10-22.

Choi, S. W., Kim*, Y. D., & Lee, G. C. (2005). Minimizing total tardiness of orders with reentrant lots in a hybrid flowshop. *International Journal of Production Research*, *43*(11), 2149-2167.

Fernandez-Viagas, V., Molina-Pariente, J. M., & Framinan, J. M. (2018). New efficient constructive heuristics for the hybrid flowshop to minimise makespan: A computational evaluation of heuristics. *Expert Systems with Applications*, *114*, 345-356.

Figielska, E. (2009). A genetic algorithm and a simulated annealing algorithm combined with column generation technique for solving the problem of scheduling in the hybrid flowshop with additional resources. *Computers & Industrial Engineering*, *56*(1), 142-151.

Figielska, E. (2014). A heuristic for scheduling in a two-stage hybrid flowshop with renewable resources shared among the stages. *European Journal of Operational Research*, *236*(2), 433-444.

Figielska, E. (2018). Scheduling in a two-stage flowshop with parallel unrelated machines at each stage and shared resources. *Computers & Industrial Engineering*, *126*, 435-450.

Guinet, A. G. P., & Solomon, M. M. (1996). Scheduling hybrid flowshops to minimize maximum tardiness or maximum completion time. *International Journal of Production Research*, *34*(6), 1643-1654.

Gupta, J. N. (1988). Two-stage, hybrid flowshop scheduling problem. *Journal of the Operational Research Society*, *39*(4), 359-364.

Hashemi-Petroodi, S. E., Dolgui, A., Kovalev, S., Kovalyov, M. Y., & Thevenin, S. (2020). Workforce reconfiguration strategies in manufacturing systems: a state of the art. *International Journal of Production Research*, 1-24.

Khalouli, S., Ghedjati, F., & Hamzaoui, A. (2010). A meta-heuristic approach to solve a JIT scheduling problem in hybrid flow shop. *Engineering Applications of Artificial Intelligence*, *23*(5), 765-771.

Khare, A., & Agrawal, S. (2019). Scheduling hybrid flowshop with sequence-dependent setup times and due windows to minimize total weighted earliness and tardiness. *Computers & Industrial Engineering*, *135*, 780-792.

Lee, G. C., & Kim, Y. D. (2004). A branch-and-bound algorithm for a two-stage hybrid flowshop scheduling problem minimizing total tardiness. *International Journal of Production Research*, *42*(22), 4731-4743.

Linn, R., & Zhang, W. (1999). Hybrid flow shop scheduling: a survey. *Computers & industrial engineering*, *37*(1-2), 57-61.

Mehravaran, Y., & Logendran, R. (2013). Non-permutation flowshop scheduling with dual resources. *Expert Systems with Applications*, *40*(13), 5061-5076.

Neufeld, J. S., Gupta, J. N., & Buscher, U. (2016). A comprehensive review of flowshop group scheduling literature. *Computers & Operations Research*, 70, 56-74.

Neufeld, J. S., Teucher, F. F., & Buscher, U. (2020). Scheduling flowline manufacturing cells with inter-cellular moves: non-permutation schedules and material flows in the cell scheduling problem. *International Journal of Production Research*, *58*(21), 6568-6584.

Öztop, H., Tasgetiren, M. F., Eliiyi, D. T., & Pan, Q. K. (2019). Metaheuristic algorithms for the hybrid flowshop scheduling problem. *Computers & Operations Research*, 111, 177-196.

Peña Tibaduiza, E., Garavito Hernández, E. A., Perez Figueredo, L. E., & Moratto Chimenty, E. (2017). Literature Review on the Hybrid Flow Shop Scheduling Problem with Unrelated Parallel Machines. *Ingeniería*, *22*(1), 9-22.

Pinedo, M. L. (2016). Scheduling: Theory, Algorithms, and Systems. Fifth Edition. Springer International Publishing.

Ribas, I., Leisten, R., & Framiñan, J. M. (2010). Review and classification of hybrid flow shop scheduling problems from a production system and a solutions procedure perspective. *Computers & Operations Research*, *37*(8), 1439-1454.

Rossit, D. A., Tohmé, F., & Frutos, M. (2018). The non-permutation flow-shop scheduling problem: a literature review. *Omega*, 77, 143-153.

Ruiz, R., & Vázquez-Rodríguez, J. A. (2010). The hybrid flow shop scheduling problem. *European Journal of Operational Research*, *205*(1), 1-18.

Ruiz-Torres, A. J., & Centeno, G. (2008). Minimizing the number of late jobs for the permutation flowshop problem with secondary resources. *Computers & operations research*, 35(4), 1227-1249.

Schaller, J., & Valente, J. M. (2019). Heuristics for scheduling jobs in a permutation flow shop to minimize total earliness and tardiness with unforced idle time allowed. *Expert Systems with Applications*, *119*, 376-386.

Wang, K., Ma, W. Q., Luo, H., & Qin, H. (2016). Coordinated scheduling of production and transportation in a two-stage assembly flowshop. *International Journal of Production Research*, *54*(22), 6891-6911.

Wörbelauer, M., Meyr, H., & Almada-Lobo, B. (2019). Simultaneous lotsizing and scheduling considering secondary resources: a general model, literature review and classification. *Or Spectrum*, *41*(1), 1-43.

Wu, C. C., Gupta, J. N., Cheng, S. R., Lin, B. M., Yip, S. H., & Lin, W. C. (2020). Robust scheduling for a two-stage assembly shop with scenario-dependent processing times. *International Journal of Production Research*, 1-16.

Yang, S., & Xu, Z. (2020). The distributed assembly permutation flowshop scheduling problem with flexible assembly and batch delivery. *International Journal of Production Research*, 1-19.

Ying, K. C., & Lin, S. W. (2018). Minimizing makespan for the distributed hybrid flowshop scheduling problem with multiprocessor tasks. *Expert systems with applications*, *92*, 132-141.

Yu, J. M., Huang, R., & Lee, D. H. (2017). Iterative algorithms for batching and scheduling to minimise the total job tardiness in two-stage hybrid flow shops. *International Journal of Production Research*, *55*(11), 3266-3282.

Table 1. Job related information.

job	$v_{j,1}$	$v_{j,2}$	y_j	d_j
j1	29	16	f1	32
j2	26	18	f2	35
j3	36	35	f1	38
j4	15	18	f2	40
j5	16	28	f1	44
j6	36	25	f1	48

Table 2. Worker related information.

worker	$e_{w,f1,1}$	$e_{w,f2,1}$	$e_{w,f1,2}$	$e_{w,f2,2}$
w1	1.00	0.70	0.70	0.95
w2	0.70	0.65	0.90	0.85
w3	0.75	0.90	0.80	1.00
w4	0.90	0.90	0.90	0.70
w5	0.80	0.80	0.85	0.80
w6	0.90	1.00	1.00	0.70
w7	1.00	0.90	0.70	0.80

Table 3. The start, duration and end times of each job for schedule S1-A1 in the AE case assumption.

	stage 1			stage 2		
job	start	duration	end	start	duration	end
j1	0.000	11.500	11.500	16.000	8.421	24.421
j2	0.000	7.761	7.761	8.000	18.000	26.000
j3	11.500	18.000	29.500	29.500	13.304	42.804
j4	7.761	4.478	12.239	26.000	15.455	41.455
j5	29.500	8.000	37.500	41.455	12.727	54.182
j6	12.239	19.487	31.725	42.804	8.929	51.732

Table 4. The completion time, the due date, and the tardiness of each job for schedule S1-A1 in the AE case assumption.

job	c_j	d_j	$tardiness$
j1	24.421	32	0
j2	26.000	35	0
j3	42.804	38	4.804
j4	41.455	40	1.455
j5	54.182	44	10.182
j6	51.732	48	3.732

Table 5. The start, duration and end times of each job for schedule S1-A1 in the SE case assumption.

	stage 1			stage 2		
job	start	duration	end	start	duration	end
j1	0.000	13.500	13.500	16.000	8.889	24.889
j2	0.000	10.000	10.000	10.000	18.000	28.000
j3	13.500	18.000	31.500	31.500	13.130	44.630
j4	10.000	5.769	15.769	28.000	14.500	42.500
j5	31.500	8.000	39.500	42.500	13.333	55.833
j6	15.769	32.423	48.192	48.192	14.175	62.368

Table 6. The completion time, the due date, and the tardiness of each job for schedule S1-A1 in the SE case assumption.

job	c_j	d_j	θ_j
j1	24.889	32	0
j2	28.000	35	0
j3	44.630	38	6.630
j4	42.500	40	2.500
j5	55.833	44	11.833
j6	62.368	48	14.368

Table 7. Summary of the scheduling rules.

Algorithm	**Step 1**	**Step 5**	**Step 6**
G1	Rule 1	Rule 1	Rule 1
G2			Rule 2
G3		Rule 2	Rule 1
G4			Rule 2
G5	Rule 2	Rule 1	Rule 1
G6			Rule 2
G7		Rule 2	Rule 1
G8			Rule 2
G9	Rule 3	Rule 1	Rule 1
G10			Rule 2
G11		Rule 2	Rule 1
G12			Rule 2

Table 8. Worker to workstation assignment per time bucket for G1.

t		**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
time scale		**0-8**	**8-16**	**16-24**	**24-32**	**32-40**	**40-48**	**48-56**	**56-64**
worker	w1	s1-2	s1-2	s1-1	s1-1	s1-1	s2-3	s1-1	--
	w2	s1-1	s1-1	s2-2	s2-2	s2-3	s2-2	--	s2-1
	w3	s1-3	s2-1	s2-1	s2-1	s2-1	s1-1	s2-1	--
	w4	s1-3	s1-3	s1-1	s1-1	s2-3	s2-2	s1-1	s2-1
	w5	s1-3	s1-1	s1-3	s1-3	s2-2	s2-3	--	--
	w6	s1-3	s1-1	s2-3	s2-3	s2-3	s2-2	s1-1	s2-1
	w7	s1-2	s1-2	s1-2	s1-2	s2-1	s1-1	s1-1	--

Table 9. Worker to workstation assignment per time bucket for G6.

t		**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
time scale		**0-8**	**8-16**	**16-24**	**24-32**	**32-40**	**40-48**	**48-56**	**56-64**
worker	w1	s1-2	s1-2	s2-1	s1-2	s1-2	s1-2	s2-1	--
	w2	s1-2	s2-2	s2-2	s2-1	s2-2	s2-1	s2-2	s2-2
	w3	s1-1	s2-1	s2-1	s2-1	s2-1	s2-2	s2-1	--
	w4	s1-1	s1-1	s1-1	s1-1	s2-2	s2-1	s2-2	s2-2
	w5	s1-2	s2-1	s1-2	s1-1	s2-1	s2-1	s2-1	--
	w6	s1-1	s1-1	s1-1	s2-2	s2-2	s1-2	s2-2	s2-2
	w7	s1-2	s1-2	s1-2	s1-2	s1-2	s1-2	--	--

Table 10. Worker to workstation assignment per time bucket for G12.

t		**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
time scale		**0-8**	**8-16**	**16-24**	**24-32**	**32-40**	**40-48**	**48-56**	**56-64**
worker	w1	s1-1	s1-3	s2-1	s1-3	s2-2	s2-1	s1-3	--
	w2	s1-3	s1-2	s1-1	s2-1	s2-2	s2-2	s2-1	s2-1
	w3	s1-2	s1-2	s2-1	s1-1	s2-2	s2-1	s2-2	--
	w4	s1-3	s1-1	s1-2	s1-1	s1-2	s2-2	s2-1	s2-1
	w5	s1-3	s1-1	s1-1	s2-1	s1-3	s2-1	s2-2	--
	w6	s1-3	s1-1	s2-2	s2-2	s2-1	s2-2	s2-1	s2-1
	w7	s1-2	s1-3	s1-3	s1-2	s1-2	s1-3	s1-3	--

Table 11. Experimental factors for the optimal solution experiments.

Factor	Level 1	Level 2
F1	$n = 10$	$n = 20$
F2	$w = 5$ $w_1^{max} = w_2^{max} = 2$ $x_1 = x_2 = 2$	$w = 10$ $w_1^{max} = w_2^{max} = 3$ $x_1 = x_2 = 3$
F3	$U(5,6)$	$U(0.5,7)$

Table 12. Results per experimental point for the optimal solution experiments.

F1	F2	F3	Number instances were all algorithms found the optimum (out of 5)	Number instances were the optimal solution was found by at least one algorithm (out of 5)	Average best tardiness for unsuccessful instances
L1	L1	L1	4	5	-
		L2	1	4	0.206
	L2	L1	3	5	-
		L2	1	1	0.075
L2	L1	L1	2	3	0.017
		L2	1	2	0.051
	L2	L1	1	5	-
		L2	0	5	-
Overall			13/40 (32.5%)	30/40 (75%)	

Table 13. Summary of results for the optimal solution experiments.

Factor	F1		F2		F3		
Level	L1	L2	L1	L2	L1	L2	Overall
% instances where all the rules found the optimum	45%	20%	40%	25%	50%	15%	32.5%
% instances where the optimal solution was found by at least one rule	75%	75%	70%	80%	90%	60%	75%

Table 14. Experimental factors for relative performance experiments.

Factor	Level 1	Level 2	Level 3
F1	$n = 30, w = 12$	$n = 50, w = 20$	-
F2	$vol_1 = vol_2 = 15$	$vol_1 = 10;\ vol_2 = 20$	$vol_1 = 20;\ vol_2 = 10$
F3	$TR = 1$	$TR = 2.5$	-

Table 15. Results by experimental variable in the AE case assumption.

	F1		F2			F3		Overall
	L1	L2	L1	L2	L3	L1	L2	
G1	0% (14.31)	0% (10.76)	0% (12.18)	0% (12.04)	0% (13.40)	0% (6.75)	0% (18.33)	0% (12.54)
G2	0% (13.71)	0% (10.41)	0% (11.83)	0% (11.78)	0% (12.56)	0% (6.35)	0% (17.77)	0% (12.06)
G3	0% (14.99)	0% (11.01)	0% (12.64)	0% (12.24)	0% (14.12)	0% (7.24)	0% (18.76)	0% (13.00)
G4	0% (14.19)	0% (10.57)	0% (12.00)	0% (11.93)	0% (13.21)	0% (6.64)	0% (18.13)	0% (12.38)
G5	30% (11.03)	15% (10.05)	18% (10.08)	24% (10.92)	26% (10.62)	18% (5.33)	27% (15.75)	23% (10.54)
G6	33% (11.04)	15% (10.04)	23% (10.01)	16% (10.93)	34% (10.68)	23% (5.31)	26% (15.77)	24% (10.54)
G7	25% (11.15)	15% (10.16)	21% (10.10)	23% (10.92)	16% (10.94)	18% (5.29)	22% (16.02)	20% (10.65)
G8	30% (11.15)	17% (10.1)	31% (10.03)	16% (10.93)	23% (10.91)	22% (5.25)	25% (15.99)	23% (10.62)
G9	9% (11.99)	11% (9.78)	11% (10.28)	11% (11.2)	8% (11.18)	7% (5.65)	13% (16.12)	10% (10.89)
G10	12% (11.96)	23% (9.77)	14% (10.25)	20% (11.2)	18% (11.15)	19% (5.61)	15% (16.12)	17% (10.86)
G11	5% (12.57)	15% (9.99)	8% (10.61)	13% (11.22)	10% (12.00)	8% (6.01)	12% (16.55)	10% (11.28)
G12	11% (12.33)	20% (9.93)	14% (10.49)	21% (11.07)	9% (11.84)	12% (5.9)	19% (16.36)	15% (11.13)

Table 16. Results by experimental variable in the SE case assumption.

	F1		F2			F3		Overall
	L1	L2	L1	L2	L3	L1	L2	
G1	3% (18.23)	8% (15.07)	4% (14.95)	3% (17.61)	9% (17.39)	3% (10.46)	7% (22.84)	5% (16.65)
G2	17% (17.37)	46% (14.08)	21% (14.35)	14% (16.69)	59% (16.13)	35% (9.61)	28% (21.84)	31% (15.72)
G3	0% (19.02)	5% (15.55)	4% (15.41)	3% (17.69)	1% (18.76)	1% (11.10)	4% (23.48)	3% (17.29)
G4	5% (17.85)	38% (14.79)	34% (14.53)	15% (16.84)	15% (17.59)	22% (10.22)	21% (22.42)	21% (16.32)
G5	14% (16.95)	3% (17.41)	9% (15.21)	5% (17.23)	11% (19.1)	6% (11.47)	11% (22.89)	8% (17.18)
G6	26% (16.93)	4% (17.11)	25% (14.96)	8% (17.22)	13% (18.87)	12% (11.29)	18% (22.74)	15% (17.02)
G7	14% (16.94)	7% (17.2)	10% (15.27)	8% (17.04)	14% (18.89)	10% (11.25)	11% (22.88)	10% (17.07)
G8	23% (16.87)	9% (16.84)	23% (15.02)	10% (16.97)	16% (18.58)	15% (11.1)	18% (22.61)	16% (16.85)
G9	11% (19.29)	7% (16.4)	4% (14.79)	19% (16.21)	4% (22.53)	7% (11.96)	11% (23.74)	9% (17.85)
G10	13% (18.69)	5% (15.74)	1% (14.51)	23% (15.83)	3% (21.31)	9% (11.32)	8% (23.11)	9% (17.22)
G11	13% (19.43)	1% (16.9)	3% (15.27)	18% (16.29)	1% (22.93)	8% (12.21)	7% (24.12)	7% (18.17)
G12	12% (18.96)	3% (16.29)	4% (14.8)	19% (16.09)	0% (21.98)	7% (11.72)	8% (23.54)	8% (17.63)

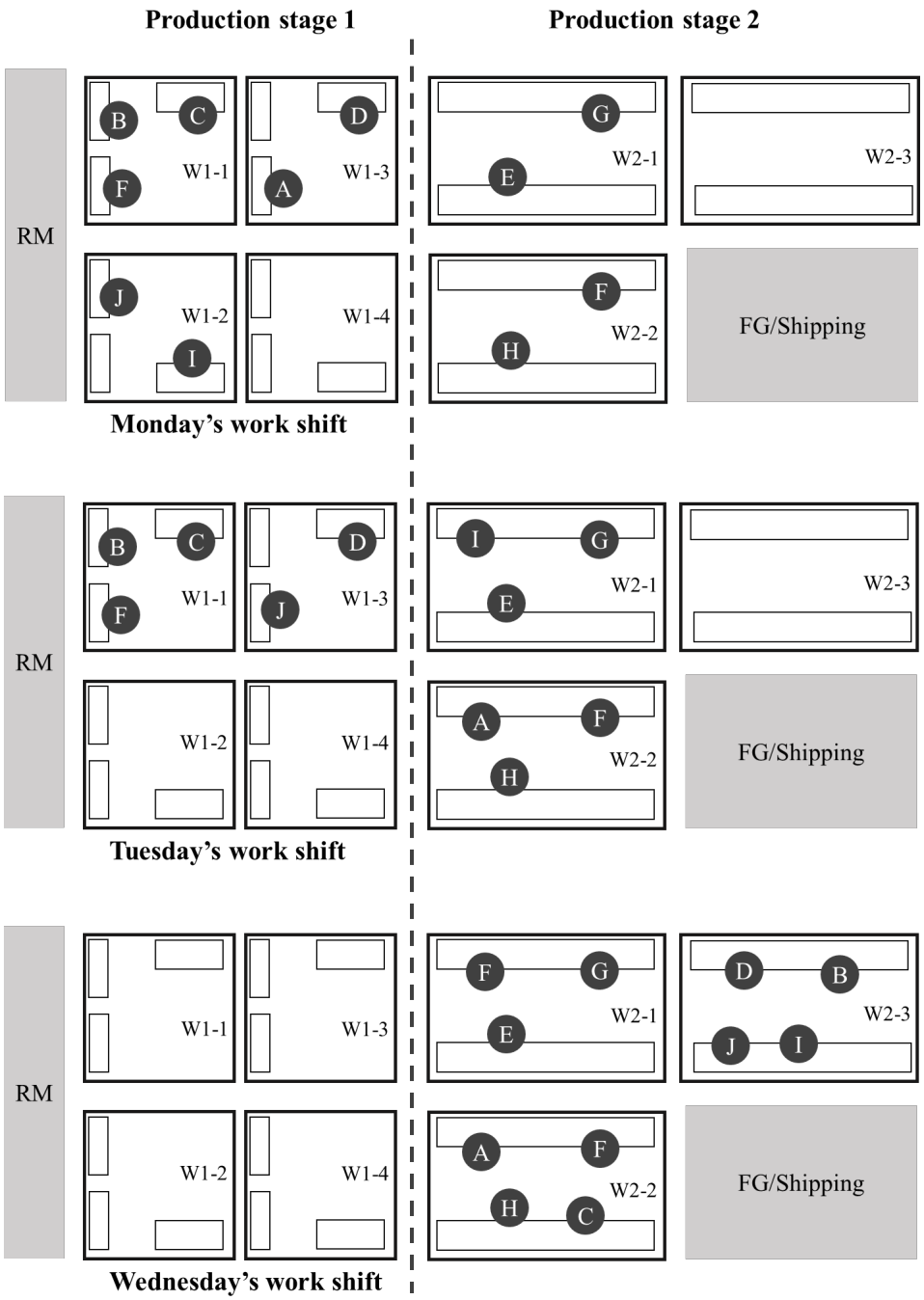

Figure 1. An illustration of the problem environment.

(i) Workstation job sequences S1

Stage 1

$K_{s1-1} = j1 - j3 - j5$

$K_{s1-2} = j2 - j4 - j6$

Stage 2

$K_{s2-1} = j1 - j3 - j6$

$K_{s2-2} = j2 - j4 - j5$

(ii) Worker assignments A1

t		1	2	3	4	5	6	7	8	...
time scale		**0-8**	**8-16**	**16-24**	**24-32**	**32-40**	**40-48**	**48-56**	**56-64**	...
worker	w1	s1-1	s1-1	s1-1	s1-1	s1-1	s2-2	s2-2	s2-2	s2-2
	w2	s1-2	s1-2	s1-2	s1-2	s2-1	s2-1	s2-1	s2-1	s2-1
	w3	s1-1	s2-2	s2-2	s2-2	s2-2	s2-2	s2-2	s2-2	s2-2
	w4	s1-2	s1-2	s2-1	s2-1	s2-1	s2-1	s2-1	s2-1	s2-1
	w5	s1-2	s1-2	s1-2	s1-2	s1-2	s1-2	s1-2	s1-2	s1-2
	w6	s1-2	s1-2	s2-1	s2-1	s2-1	s2-1	s2-1	s2-1	s2-1
	w7	s1-1	s1-1	s1-1	s1-1	s1-1	s2-2	s2-2	s2-2	s2-2

AE Case

(iii) Schedule resulting from combining S1 and A1

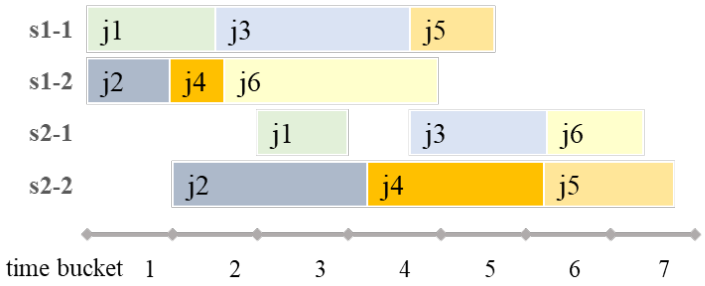

Figure 2. Schedule based on job sequences S1 and worker assignments A1 in the AE case assumption.

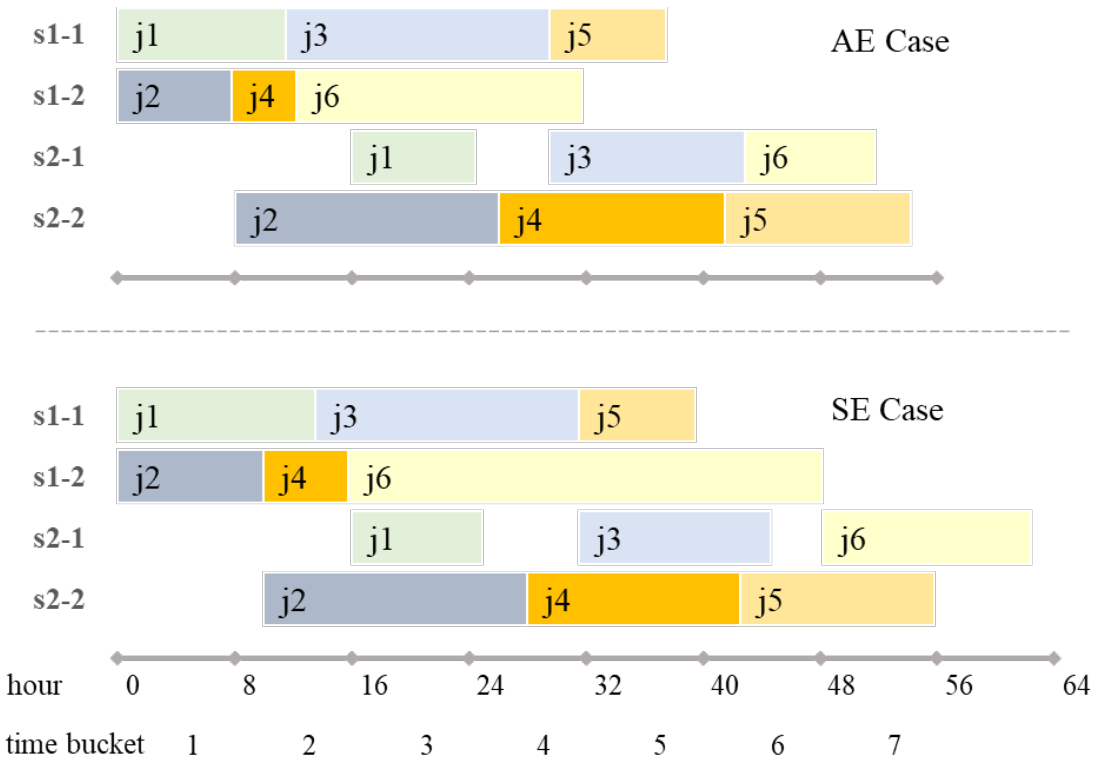

Figure 3. Schedules based on job sequences S1 and worker assignments A1 in the AE and SE case assumptions.

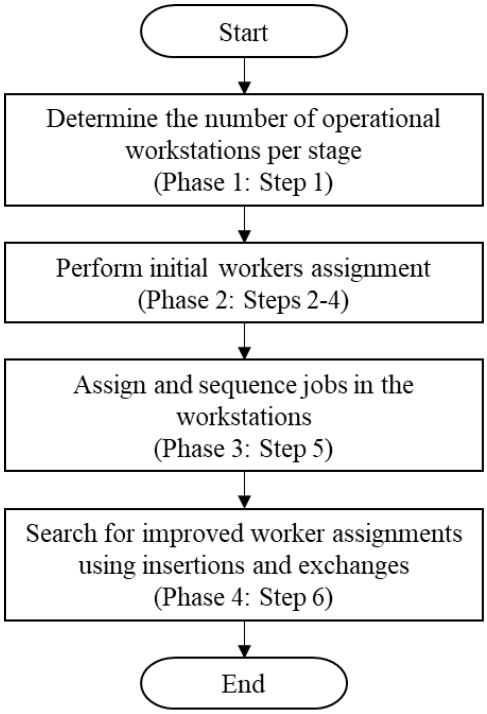

Figure 4. Flowchart of the solution approach.

Figure 5. Schedules resulting in the implementation of algorithms G1, G6, and G12 in the SE case.