Computer Systems Science & Engineering
DOI:10.32604/csse.2022.018259
Article

Tech Science Press

Heuristic Scheduling of Job Orders in a Build-to-Order Manufacturing System

Chia-Nan Wang[1], Chien-Chang Chou[2], Yu-Chi Chung[1,*], Nguyen Ky Phuc Phan[3], Van Thanh Nguyen[4] and Viet Tinh Nguyen[4]

[1]Department of Industrial Engineering and Management, National Kaohsiung University of Science and Technology, Kaohsiung, 80778, Taiwan
[2]Department of Shipping Technology, National Kaohsiung University of Science and Technology, Kaohsiung, 80778, Taiwan
[3]Faculty of Industrial Engineering and Management, International University, Ho Chi Minh City, 70000, Vietnam
[4]Faculty of Commerce, Van Lang University, Ho Chi Minh City, 70000, Vietnam
*Corresponding Author: Yu-Chi Chung. Email: ycchung@nkust.edu.tw
Received: 03 March 2021; Accepted: 02 May 2021

Abstract: With the continuous development of technology, traditional manual work has been becoming more and more automated. Most large or medium-sized companies have applied Enterprise Resource Planning (ERP) software into their business and production activities. However, since many small firms cannot afford ERP because of its expensive cost, they often still employ manual work for the same tasks this software resolves, especially for scheduling. This paper aims to provide a possible solution for small businesses to try automated scheduling and discover whether it can help much. There are two main ways to make this determination: a mathematical model and a heuristic model, which are suitable for assessing low- and medium-sized workloads, respectively. This case study was carried out in a small domestic interior furniture company, particularly in scheduling for their customized products in two-stage flow shop. Normally, they produce according to the sequence of customers' orders. However, when we applied these supportive tools with batch-processing machines, they experienced enhanced production performance due to diminishing setup time for distinctive items and a more streamlined arrangement of job sequences. These changes were implemented for some small companies that do not use many production stages and have a suitable number of jobs and customers. If this method were applied to larger demands, it would need further improvement and development to become a complete tool that can perform like a part of an ERP system.

Keywords: scheduling; mathematical; heuristic; flow shop; batch-processing

1 Introduction

Scheduling is a decision-making process, which is extensively used in manufacturing and service industries, that works to allocate resources to jobs in the given time in order to optimize certain goals. An enterprise's resources and tasks can be unique, such as machines on the shop floor and operations in a production process. Each task can be prioritized and noted with a release date as well as a due date. In

This work is licensed under a Creative Commons Attribution 4.0 International License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

this way, the objective is the minimization of the completion time or minimization of the number of tasks needed to carry out a given task. For example, one specific objective of scheduling systems is to minimize the sum of the penalties due to late delivery or a lack of goods. There are many approaches to manufacturing problems and each approach is characterized by its method and implementation.

A build-to-order (BTO) strategy combines the features and strengths of both make-to-stock (forecast driven) and make-to-order (demand-driven) strategies. Schedulers employing BTO often face the problem of customer orders in mass customizing. Salespeople are flexible to limit numerous choices for customers and publish the suggestion in catalogs in order to help schedulers more easily to combine a variety of styles with a wide range of fabric types to cut down the setup time and the number of different manufacturing stages for the firms. However, there are some deeper issues in need of further consideration, such as the potential effectiveness of assigning a batch to an available, parallel machine.

The research of this study focuses on the scheduling problem inherent to the BTO manufacturing strategy in a two-stage flow shop with batch-processing machines. Three main methodologies are used: mixed-integer linear programming, fuzzy logic, and a genetic algorithm. More specifically, a mixed-integer linear programming method is combined with fuzzy logic to create a fuzzy linear programming model, which can help to minimize the flow time more effectively. In addition, the genetic algorithm is applied to search for a set of optimal solutions over iterations from the initial population. The result from the scheduling model is then applied into the real system in order to achieve better productivity. The result of this research can be expanded and applied to scheduling problems with similar manufacturing environment and setups.

2 Literature Review

This part of the research reviews the literature on scheduling with batching according to the family of jobs with the same setup, using two primary methods, which are mixed-integer linear programming and the heuristic algorithm. Linear programming maximizes or minimizes a linear objective function subject to one or more constraints. Mixed-integer programming (MIP) adds one additional condition that at least one of the variables can only take on integer values. In other words, an optimization model is an integer program; if any of its decision variables are not discrete, the model is a mixed-integer program. MIP models have other types such as mixed-integer quadratic programming (MIQP) with a quadratic objective but without quadratic constraints and mixed-integer quadratically constrained programming (MIQCP), which do have quadratic constraints. The basic mixed-integer programming with any quadratic features is often referred to as a mixed-integer linear programming (MILP) problem.

A heuristic algorithm is a procedure used to determine near-optimal solutions for an optimization problem. In this case, which considers a minimum makespan problem, the heuristic is a distinct method developed to address a specific optimization problem. A simple heuristic to solve this problem is the list heuristic, in which jobs are listed out in some given order, and then each job in the list is performed by one machine at a time. This process is continued until every job is completed; the completion time of machines corresponds to the processing time of jobs.

Many scheduling models propose to deal with a variety of manufacturing scheduling problems, though their different features require them to be adjusted before each case-based application [1–7]. The flow shop problem was studied by Johnson [8] for two machines, whose research was then developed by Gonzalez et al. [9] to consider cases involving more than two machines. However, there still seems to be only a limited number of models that can deal with the specific situation of batch processing machines in a flow shop. In these kinds of cases, Potts et al. [10] offer several reasons to arrange jobs into batches. Jobs can be grouped and performed continuously if they have the same setup on machines. Another way to batch jobs with the same processing time is to assign them simultaneously as a batch using several machines in a

single stage [11]. Based on the properties of each scheduling problem, an optimal method can be chosen to develop an optimal production schedule strategy.

Mixed-integer programming has become a common approach to schedule batch-processing machines; hence, there are several papers covering this topic. For example, as Damodaran et al. [12] propose, when dealing with scheduling batches of jobs on two machines in a flow shop, mixed-integer formulations can be used to solve the problem and prescribe optimal solutions with zero or unlimited buffer capacity. To check the validity of their formulations, they used commercial solvers such as CPLEX and OSL to apply branch and bound, believing that good lower bounds could aid the solvers in proving optimal solutions in a shorter period of time. Yimer et al. [13] also applied mixed-integer programming in their case study about scheduling job orders in a two-stage flow shop with batch-processing machines in the mass customizing furniture industry. They noted that a machine can carry out one job at a time and cannot perform any other processes during the time in which it is preparing or being set up. The constraints in their mathematical model fulfill the due date quest and minimize the total weighted flowtime. More specifically, the customers in their case study each had their own priority level with the considered company; hence, this factor was also included as a constraint. Moreover, they expanded on Damodaran and Srihari's case [12], with an additional implementation, while considering the fuzzy technique to make sure that the imprecision associated with estimation of setup and processing times was minimal.

A heuristic algorithm is often used as an alternative method in accompaniment with the mathematical model, especially in mixed-integer programming, to gain the result for comparison with the optimal solution [14–21]. Bhongade et al. [22] studied the assembly flow shop in which some jobs had no processes running on one or more machines. To obtain a near optimal solution, they used heuristics to calculate the makespan for all possible alternatives and also determined the lower and upper bound values of the makespan at each step to branch off the least makespan. Their work discovered that heuristics should be applied according to initial goals to achieve a better performance and can be extended to improve the solution using metaheuristics, such as a genetic algorithm (GA), which was also used by Yimer et al. [13] to find an alternative solution, compared with the one resulting from mixed-integer programming. Additionally, Gupta et al. [23] attempted to develop a simple heuristic without much satisfaction about the resulting accuracy of the optimal solution; however, their work provides an initial solution as fast as possible while weighting other methods to solve the flow shop scheduling problems to minimize the makespan. More specifically, they developed their heuristic based on the reduced weighted scheme of machines at every stage to create a distinctive combination of sequences for producing optimal results. They used the heuristic to test various benchmark problems and selected the best sequence with the minimum makespan. Their research proved that when the number of jobs increased, their heuristic still provided good quality results.

The above literature review reveals a great deal of information that has inspired others to continue researching further methods to solve for flow shop scheduling problems. In this research, a fuzzy mixed-integer programming and a heuristic algorithm will be implemented to propose a model for solving the scheduling problems in a BTO manufacturing strategy in a two-stage flow shop with batch-processing machines.

3 Research Graph

The MIFLP model is built based on the issue of the case study, which is the need to solve the scheduling problem for jobs on parallel machines in a two-stage process. The objective is to determine the set of jobs included in each family and to sequence the batches in order to minimize the total weighted flow time by reducing the setup time for a group of jobs. For the small size, CPLEX was used to code the MIFLP, while MATLAB was used to code the proposed heuristic algorithm, which can give a higher quality

solution with medium computational effort for larger sizes. The model development procedure was carried out in four phases, as shown in Fig. 1.

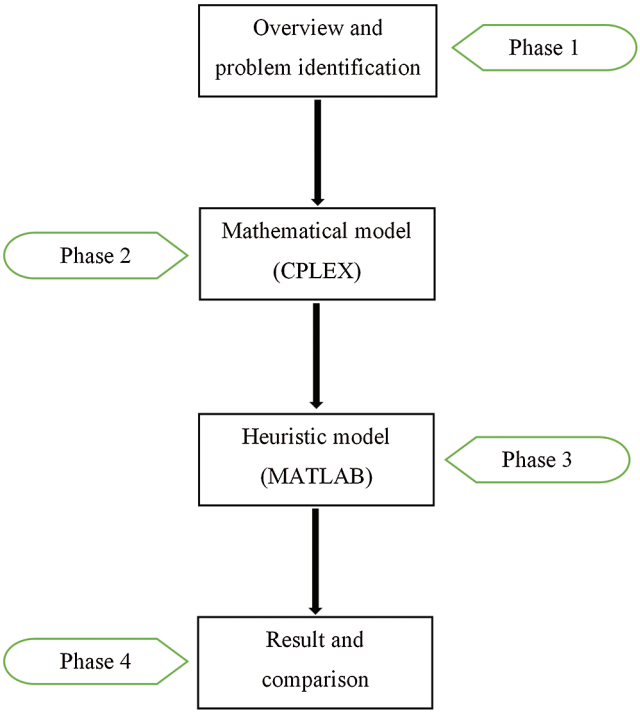

Figure 1: Research graph

Phase 1: Identify the problem, collect related data, and analyze the current manufacturing processes.

Phase 2: Based on the information collected about characteristics of the manufacturing process and the raw data from the company, develop a mathematical model with some typical constraints. Then, convert the model into CPLEX programming with the support of tuple approach and apply the analyzed data into it to test the validity of the model.

Phase 3: Based on the manual solution to the problem, develop the heuristic model with the flow of the manual calculation. The inclusion of additional data does not appear to affect computational time, which remains the same. Although this may be not an optimal solution, it can be the fastest and most reasonable method to obtain an optimal schedule.

Phase 4: Both models are tested against same-size problems and the results are then analyzed and compared to obtain a recommendation for a suitable solution for the company.

4 Solution Approach

4.1 Mathematical Model

In a standard mathematical model of a two-stage flow shop scheduling with batch-process machines, there are several indices that should be included, such as jobs, customers, groups, stages, and machines. The output of this model is covered by some decision variables, including the combination of jobs into a batch, starting time, and processing time, as well as completion time of each batch, completion time of a customer, and the makespan.

4.1.1 Notations

The notations for indices and sets are shown in Tab. 1 below:

Table 1: Notations for indices and sets

Notations	Indices and sets
j	Index of jobs, j = 1, …, J
k	Index of customers, k = 1, …, K
i	Index of processing stages, i = 1, …, I
g	Index of job families or groups, g = 1, …, G
b	Index of processing batches, b = 1, …, B
t	T = J*I*B, t = 1, …, T
M	Positive number
Ω_k	Set of job indices from customer k
Ω_g	Set of job indices in group g

The notations for parameters are shown in Tab. 2 below:

Table 2: Notations for parameters

Notations	Parameters
$W_{k,j}$	1 if job j is ordered by customer k ($j \in \Omega_k$), or 0 otherwise
$Z_{g,j}$	1 if job j is member of g_i ($j \in \Omega_g$), or 0 otherwise
r_k	Release time for orders made by customer k
d_k	Due date for orders made by customer k
m_i	Number of available machines at stage i
$a_{g,i}$	Machine setup time at stage i for jobs in group g
$p_{j,i}$	Processing time of job j at stage i

The notations for variables are shown in Tab. 3 below:

Table 3: Notations for variables

Notations	Variables	Condition
$X_{b,j}$	1 if job j is assigned to batch b ($j \in \Omega_b$), or 0 otherwise	
$Y_{g,b}$	1 if all jobs into batch b are from group g ($\Omega_b \subseteq \Omega_g$), or 0 otherwise	
n_b	Number of jobs assigned to batch b	$n_b \geq 0$
$c_{j,i}$	Completion time of job j at stage i	$cj_{j,i} \geq 0$
$sbat_{b,i}$	Processing start time of first job in batch b at stage i	$sbat_{b,i} \geq 0$
$pbat_{b,i}$	Processing time of all jobs in batch b at stage i	$pbat_{b,i} \geq 0$

(Continued)

Table 3 (continued).

Notations	Variables	Condition
$cbat_{b,i}$	Completion time of last job in batch b at stage i	$cbat_{b,i} \geq 0$
$ccus_k$	Completion time of last job at final stage of customer k	$ccus_k \geq 0$
cmax	Imprecise makespan	cmax $\geq$ 0
F_b	Number of groups assigned to a batch	$F_b \geq 0$
$U_{b,i,t}$	1 if time t $\geq$ processing start time of first job in batch b at stage i	$U_{b,i,t} \geq 0$
$V_{b,i,t}$	1 if completion time of last job in batch b at stage i $\geq$ time t	$V_{b,i,t} \geq 0$
$R_{b,i,t}$	1 if completion time of last job in batch b at stage i $\geq$ time t $\geq$ processing start time of first job in batch b at stage i	$R_{b,i,t} \geq 0$

4.1.2 Model Development

The objective is to minimize the makespan in order to use the resources optimally and meet the due date of all of the customers. The general form includes the below assumptions:

- Jobs are independent and assigned to groups in terms of the features of the machines' setup time
- The release date and due date of jobs are dependent on the customers
- There is no priority between customers; hence, all jobs have a similar priority, which means they are equally considered
- In the flow shop, there are two stages in which the setup time for each group and the processing time for each job are different
- All machines in two stages are available from the beginning of considered period (supposed day 0)
- Break downs are not considered in this model.

The objective function is to minimize the imprecise makespan (cmax).Subject to:

$$\sum_{b=1}^{B} X_{b,j} = 1 \forall j \tag{1}$$

Constrain (1) ensures that a job is assigned to exactly one batch.

$$\sum_{g=1}^{G} Y_{g,b} \leq 1 \ \forall b \tag{2}$$

Constrain (2) restricts that all jobs assigned to a batch are derived from the same group.

$$X_{b,j} \leq Z_{g,j} \ \forall bj \in \Omega g \tag{3}$$

$$X_{b,j} \leq Y_{g,b} \ \forall bj \in \Omega g \tag{4}$$

Constraint (3) and (4) control that a job in a given group can be assigned a batch if and only of the group itself is assigned to the batch.

$$\mathrm{nb} \;=\; \sum_{j=1}^{J} \mathrm{X_{b,j}} \;\; \forall b \tag{5}$$

Constraint (5) determines the number of jobs assigned to a batch.

$$\mathrm{Fb} \;=\; \sum_{g=1}^{G} \mathrm{Y_{g,b}} \;\; \forall b \tag{6}$$

$$\mathrm{sbat_{b,i}} \;\; \leq \;\; \mathrm{F_b}^{*}\, \mathrm{M} \; \forall b \; \forall i \tag{7}$$

$$\mathrm{cbat_{b,i}} \leq \; \mathrm{F_b}^{*} \mathrm{M} \; \forall b \; \forall i \tag{8}$$

Constraint (6), (7), and (8) control which batch is continued to be calculated.

$$\mathrm{sbat_{(b,1)}} \;\geq \mathrm{r_k} \;* \mathrm{X_{b,j}} + \left(\mathrm{X_{b,j}} - 1\right) * M \; \forall b \; k,j \;\in \Omega_{\mathrm{k}} \tag{9}$$

Constraint (9) ensures that the processing start time of the first job in batch b at stage 1 is not less than the released time of all jobs in the batch.

$$\mathrm{sbat_{b,i}} \;\geq \mathrm{c_{j,i-1}} \;+\; \left(\mathrm{X_{b,j}} - 1\right) * M \;\; \forall b \; \forall j \; (2 \;\leq \;\; i \;\leq \mathrm{I} \;-\; 1) \tag{10}$$

Constraint (10) ensures that processing start time of the first job in batch b at stage i with $i \geq 2$ is not less than the completion time of jobs belonging to that batch in the previous stage.

$$\mathrm{sbat_{b,i}} \;\geq \mathrm{cbat_{b,i-1}} \;+\; (\mathrm{F_b} - 1) * M \;\; \forall b \; (\mathrm{i} \;\geq \;\; 2) \tag{11}$$

Constraint (11) ensures that processing start time of the first job in batch b at stage i with $i \geq 2$ is not less than the completion time of that batch in the previous stage.

$$\mathrm{pbat_{b,i}} \;\; = \sum_{g=1}^{G} \mathrm{a_{g,i}} \;* \mathrm{Y_{g,b}} \;+\; \sum_{j=1}^{J} \mathrm{p_{j,i}} \;*\; \mathrm{X_{b,j}} \; \forall i \; \forall b \tag{12}$$

Constraint (12) determine the batch processing time period required at each stage (the setup time for machines at that stage is considered as a part of processing time to make easier for calculation).

$$\mathrm{cbat_{b,i}} \;\geq\; \mathrm{sbat_{b,i}} \;\; + \; \mathrm{pbat_{b,i}} \;\; - \;\; 1 \;\; + \;\; (\mathrm{F_b} - 1) * M \; \forall b \; (\mathrm{i} \;\; = \;\; 1) \tag{13}$$

$$\mathrm{bat_{b,i}} \;\geq\; \mathrm{sbat_{b,i}} \;\; + \; \mathrm{pbat_{b,i}} \;\; + \; (\mathrm{F_b} - 1) * M \; \forall b \; (\mathrm{i} \;\geq \;\; 2) \tag{14}$$

Constraint (13) and (14) determine the completion time of last job in batch b at stage i.

$$\mathrm{c_{j,i}} \;\geq \mathrm{cbat_{b,i}} \;+\; \left(\mathrm{X_{b,j}} - 1\right) * M \;\; \forall i \; \forall b \; \forall j \tag{15}$$

Constraint (15) ensures that completion time of job j belonging to batch b at stage i must be not less than completion time of that batch at that stage.

$$\mathrm{ccus_k} \;\geq\; \mathrm{c_{j,i}} \;*\; \mathrm{W_{k,j}} \;\; k,j \;\in\; \Omega_{\mathrm{k}} \;\; i = I \tag{16}$$

Constraint (16) determines the completion time for set of jobs from the same customer k by the latest completion time of that set.

$$\mathrm{ccus_k} \;\leq\; \mathrm{d_k} \;\; \forall k \tag{17}$$

Constraint (17) restricts the completion time for set of jobs of a customer within the promised due date.

$$\text{cmax} \geq \text{ccus}_k \quad \forall k \tag{18}$$

Constraint (18) determines the makespan.

$$t \geq \text{sbat}_{b,i} - 1 - \left(1 - \text{U}_{b,i,t}\right) * M - (1 - \text{F}_b) * M \; \forall b \; \forall i \; \forall t \tag{19}$$

$$t \leq \text{sbat}_{b,i} - 1 + \text{U}_{b,i,t} * M + (1 - \text{F}_b) * M \; \forall b \; \forall i \tag{20}$$

$$t \leq \text{cbat}_{b,i} + 1 + \left(1 - \text{V}_{b,i,t}\right) * M + (1 - \text{F}_b) * M \; \forall b \; \forall i \; \forall t \tag{21}$$

$$t \geq \text{cbat}_{b,i} + 1 - \text{V}_{b,i,t} * M - (1 - \text{F}_b) * M \; \forall b \; \forall i \; \forall t \tag{22}$$

$$\text{U}_{b,i,t} \leq \text{F}_b \; \forall b \; \forall i \; \forall t \tag{23}$$

$$\text{V}_{b,i,t} \leq \text{F}_b \; \forall b \; \forall i \; \forall t \tag{24}$$

$$\text{R}_{b,i,t} \leq \text{U}_{b,i,t} \quad \forall b \; \forall i \; \forall t \tag{25}$$

$$\text{R}_{b,i,t} \leq \text{V}_{b,i,t} \; \forall b \; \forall i \; \forall t \tag{26}$$

$$\text{R}_{b,i,t} \geq \text{U}_{b,i,t} + \text{V}_{b,i,t} - 1 \; \forall b \; \forall i \tag{27}$$

$$\sum_{b=1}^{B} \text{R}_{b,i,t} \leq \text{m}_i \; \forall i \; \forall t \tag{28}$$

Constraint (19), (20), (21), (22), (23), (24), (25), (26), (27) and (28) ensure that the number of batches assigned at the time t is not greater than number of available machines.

4.2 Heuristics Model

Similar to the mathematical model above, the heuristic model also needs several indices, as mentioned, but they need to be specially organized in the data structure to make it easier to check the flow of information. Such an algorithm is often more efficient than employing a mathematical model to find a reasonable solution with a low computational time. It combines both computation and programming to achieve the output of this model, which is to minimize the makespan.

4.2.1 Data Structure

There are six objects taken into consideration for execution: job, customer, group, machine, stage, and batch. Before launching the proposed method, the data structure of each object has to be carried out to satisfy the objectives. The data structure of each object is given in the tables below (Tabs. 4–10):

Table 4: Data structure of jobs

Parameter	Description
Number	Job's number
Protime	Processing time of a job in two stages
Customer	The customer who orders the job
Group	The group in which a job is assigned
Batch	The batch in which a job is assigned

Table 4 (continued).

Parameter	Description
Release	The release date of a job depends on the release date of customer who orders that job
Completion	The completion time of a job is the completion time of the batch in which that job is contained
Due	The due date of a job depends on the due date of customer who orders that job
Tardiness	The tardiness of a job to consider whether the job is late

Table 5: Data structure of customers

Parameter	Description
Number	Customer's number
Jobsequence	The jobs which are ordered by a customer
Release	The release date of customer
Completion	The completion time of a customer
Due	The due date of a customer
Tardiness	The tardiness of a customer to consider whether the customer is late

Table 6: Data structure of groups

Parameter	Description
Number	Group's number
Setup	Setup time of a group in two stages
Batsequence	The batches which belong to a group

Table 7: Data structure of machines

Parameter	Description
Number	Machine's number
Stage	The stage to which the machine belongs
Batsequence	The batch sequence which goes through the machine
Release	The release time of the machine
Start	The start time of the machine
Completion	The completion time of the machine
Idletime	The idle time of the machine
Curbat	The current batch in the machine

Table 8: Data structure of stage

Parameter	Description
Number	Stage's number
Maxrelease	The max release time of a stage
Maclist	The machine list contains machines in the stage

Table 9: Data structure of batches

Parameter	Description
Number	Batch's number
Group	The group which batch belongs to
Jobsequence	The jobs belong to a batch
Release	The release time of the batch
Start	The start time of the batch
Process	The processing time of the machine
Completion	The completion time of the batch
Macsequence	The machine sequence that a batch go through
Stasequence	The stage sequence that a batch go through
Cursta	The current stage which the batch at
Flag	The completion flag to mark done batch

4.2.2 Coding Diagram

By using MATLAB as an optimization program to implement the proposed heuristic, the steps when adapting this method is shown below:

0. Input data information from excel.
1. Transfer the input into needed form (data structure).
2. Assign jobs into batch by assigning all jobs in group into a similar batch.
3. Calculate the processing time of batches at each stage and the completion time of batches for both stages:

   For b = 1 number of batches

   {

   Processing time of batch = setup time of group including the considered batch + processing time of all jobs in considered batch

   Completion time of batch = total processing time of batch in both stages

   }

   End For
4. Split the batches:

   For e = 1 number of batches – number of groups

{

Choose the batch having highest completion time

Split the batch into 2 batches which both of them have the completion time about that of the initial batch

Update the batch number: one keeps the number of the initial batch, and other has the number = the number of the initial batch + e

Update the processing time of batches at each stage as well as the completion time of batches for both stages

}

End For

5. Update the jobs which are assigned into a batch, and processing time of that batch as well as the release time of the batch which is the latest release date of all jobs in that batch
6. Update the batches belong to a group.
7. Assign batches into machines.

For i = 1 number of stages

{

For b = 1 number of batches

{

When the current stage of batch b = i and the completion time of batch b is not completed

Choose the machine m have the minimum release time in stage i to assign the batch into

Starting time = max (release time of machine m, release time of batch b)

When batch b belongs to the same group to which the previous batch is processed in machine m

{

Starting time = Starting time – setup time of the group that batch b belongs to

}

Completion time = starting time + processing time of batch in stage i

If the current stage of batch b is the last stage

The completion time of batch b is completed

Else

Update the next stage that batch b will go through

End If

}

End For

}

End For

8. Calculate the completion time and tardiness of jobs as well as customers.
9. Calculate the idle time of each machine and plot the processing time on machines figure.

5 Case Study

The proposed mathematical model and heuristics model are tested against a real-world manufacturing system scheduling problem. Then, the results, as well as the efficiency of the two models, are analyzed and compared to decide which model is the optimal tool for the given scheduling problem.

5.1 Results Comparison

The result of some typical trials is shown below to make comparisons:

Table 10: Result comparison

Problem size	Method	Cmax	Computation time
10 jobs (4 machines)	Mathematical model (CLPEX)	55	15 seconds
	Heuristic model (MATLAB)	57	23 seconds
20 job (4 machines)	Mathematical model (CLPEX)	87	1 hour 17 seconds
	Heuristic model (MATLAB)	95	23 seconds
30 jobs (4 machines)	Heuristic model (MATLAB)	138	24 seconds
100 jobs (6 machines)	Heuristic model (MATLAB)	239	24 seconds

5.2 Result Analysis

When the number of jobs is small (in this case, the number of jobs is 10 or 20), it is clear that the results of the mathematical model in the CPLEX program can achieve a more optimal solution. However, in reality, the number of jobs can be significantly higher than 20, which means it is essential to have a serious scheduling strategy in order to minimize the idle time of the machines, as well as to minimize the makespan and meet the customer due dates. The computation time for cases with 20 or more jobs is very high for the mathematical model in comparison to the heuristic model.

Other trials with 30 jobs and 4 machines (in case of rush orders, especially from August to January when the demand is dramatically higher) and with 100 jobs and 6 machines (for example, when the company plans to expand their capacity to meet the expected demand in the future), the MATLAB program with the heuristic model obtains the solution in a few seconds, although this solution may not be the optimal one.

Therefore, if the size of problem is small (under 20 jobs), the mathematical model in the CPLEX program would be better suited for use in obtaining an optimal solution. By contrast, larger sized problems (more than 20 jobs) require a scheduling problem to be solved using the heuristic model in MATLAB in order to obtain the schedule faster and with a relatively moderate level of efficiency.

6 Conclusion

This research focuses on solving scheduling of job orders in a two-stage flow shop using two primary methods: a CPLEX program with a mathematical model and MATLAB with a heuristic model. Although CPLEX always achieves the optimal solution, MATLAB is preferred when the size of problem crosses a certain size threshold. MATLAB's computation time is also more acceptable than CPLEX for solving larger problems.

Under the given conditions, manufacturing performance was shown to be enhanced clearly when compared to the time taken to manually schedule the same problem. First, the setup time was reduced

due to a manual grouping process based on worker experience. Second, the arrangement of batches into machines by heuristic algorithm helped reduce the idle time of the machines as well as reduce the associated waste of energy and time. Due to this savings in time, the company can improve the capacity to receive more orders and ultimately achieve a higher margin for profit.

The current model is good enough for the two-stage flow shop to solve the problem. The model can also handle three-stage or four-stage flow shop problems, although only those of a small size. To have a more practical application in the future, this model needs to be developed to perform with more constraints, especially in consideration of jobs with an automated group process instead of a manual one. Moreover, in addition to the heuristic algorithm, others such as a genetic algorithm should be developed and compared to the proposed model to make the comparison for the most suitable decision. Additionally, GUI programming should be implemented to achieve a more user-friendly application for operators. Finally, this research does not consider all of the possible impacts of the working environment, such as machine breakdowns or customer order changes, which should be included in further and deeper research.

Funding Statement: The authors received no specific funding for this study.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] M. Gen, L. Lin, C. Y. Hsu, C. F. Chen, T. Borangiu *et al.,* "Multiobjective evolutionary algorithm for manufacturing scheduling problems: State-of-the-art survey," *Journal of Intelligent Manufacturing*, vol. 25, no. 5, pp. 849–866, 2014.

[2] T. Chen and Y. C. Wang, "A nonlinear scheduling rule incorporating fuzzy-neural remaining cycle time estimator for scheduling a semiconductor manufacturing factory—a simulation study," *International Journal of Advanced Manufacturing Technology*, vol. 45, no. 1–2, pp. 110–121, 2009.

[3] W. Xiang and H. P. Lee, "Ant colony intelligence in multi-agent dynamic manufacturing scheduling," *Engineering Applications of Artificial Intelligence*, vol. 21, no. 1, pp. 73–85, 2008.

[4] W. Shen, "Distributed manufacturing scheduling using intelligent agents," *IEEE Intelligent Systems*, vol. 17, no. 1, pp. 88–94, 2002.

[5] K. Ding, P. Jiang and M. Zheng, "Environmental and economic sustainability-aware resource service scheduling for industrial product service systems," *Journal of Intelligent Manufacturing*, vol. 28, no. 6, pp. 1303–1316, 2015.

[6] K. Lenin, "Hybridization of genetic particle swarm optimization algorithm with symbiotic organisms search algorithm for solving optimal reactive power dispatch problem," *Journal of Applied Science, Engineering, Technology, and Education*, vol. 3, no. 1, pp. 12–21, Jun. 2020.

[7] A. Handayani and C. Y. Setyatama, "Analysis of supply chain management performance using SCOR and AHP methods in green avenue apartments of East Bekasi," *Journal of Applied Science, Engineering, Technology, and Education*, vol. 1, no. 2, pp. 141–148, Jan. 2020.

[8] S. M. Johnson, "Optimal two and three-stage production schedules with set-up times included," *Naval Research Logistics Quarterly*, vol. 1, pp. 61–68, 1953.

[9] T. Gonzalez and S. Sahni, "Flow shop and job shop schedules: Complexity and approximation," *Operations Research*, vol. 26, no. 1, pp. 36–52, 1978.

[10] C. N. Potts and M. Y. Kovalyov, "Scheduling with batching: A review," *European Journal of Operational Research*, vol. 120, no. 2, pp. 228–249, 2000.

[11] V. Chandru, C. Y. Lee and R. Uzsoy, "Minimizing total completion time on a batch processing machine with job families," *Operations Research Letters*, vol. 13, no. 2, pp. 61–65, 1993.

[12] P. Damodaran and K. Srihari, "Mixed-integer formulation to minimize makespan in a flower shop with batching processing machines," *Mathematical Computer Modelling*, vol. 40, pp. 1465–1472, 2004.

[13] A. D. Yimer and K. Demirli, "Fuzzy scheduling of job orders in a two-stage flowshop with batch-processing machines," *International Journal of Approximate Reasoning*, vol. 50, no. 1, pp. 117–137, 2009.

[14] T. Chen and G. Zhou, "Modeling production scheduling problem and its solution by genetic algorithm," *Journal of Computers*, vol. 8, no. 8, pp. 2126, 2013.

[15] U. P. Wen and C. I. Yeh, "Tabu search methods for the flow shop sequencing problem," *Journal of the Chinese Institute of Engineers*, vol. 20, no. 4, pp. 465–470, 1997.

[16] E. Jiang and L. Wang, "An improved multi-objective evolutionary algorithm based on decomposition for energy-efficient permutation flow shop scheduling problem with sequence-dependent setup time," *International Journal of Production Research*, vol. 57, no. 6, pp. 1756–1771, 2019.

[17] T. H. Tran and K. M. Ng, "A hybrid water flow algorithm for multi-objective flexible flow shop scheduling problems," *Engineering Optimization*, vol. 45, no. 4, pp. 483–502, 2013.

[18] M. B. Fakhrzad and M. Heydari, "A heuristic algorithm for hybrid flow-shop production scheduling to minimize the sum of the earliness and tardiness costs," *Journal of the Chinese Institute of Industrial Engineers*, vol. 25, no. 2, pp. 105–115, 2008.

[19] M. Gen, M. K. Tiwari, P. C. Chang and C. J. Liao, "Meta-heuristics for manufacturing scheduling and logistics problems," *International Journal of Production Economics*, vol. 141, no. 1, pp. 1–3, 2013.

[20] O. T. Baruwa and M. A. Piera, "Anytime heuristic search for scheduling flexible manufacturing systems: a timed colored petri net approach," *International Journal of Advanced Manufacturing Technology*, vol. 75, no. 1, pp. 123–137, 2014.

[21] S. E. Kesen, S. K. Das and G. Zulal, "A genetic algorithm based heuristic for scheduling of virtual manufacturing Cells (VMCs)," *Computers & Operations Research*, vol. 37, no. 6, pp. 1148–1156, 2010.

[22] A. Bhongade and P. Khodke, "A heuristic for production scheduling problem with machining and assembly operations," *International Journal of Industrial Engineering Computations*, vol. 3, no. 2, pp. 185–198, 2012.

[23] A. Gupta and S. R. Chauhan, "A heuristic algorithm for scheduling in a flow shop environment to minimize makespan," *International Journal of Industrial Engineering Computations*, vol. 6, no. 2, pp. 173–184, 2015.