International Journal of Technology 11(3) 608-618 (2020)
Received September 2019 / Revised January 2020 / Accepted May 2020

International Journal of Technology

http://ijtech.eng.ui.ac.id

A Batch Scheduling Model for a Three-stage Hybrid Flowshop Producing Products with Hierarchical Assembly Structures

Rahmi Maulidya[1*], Suprayogi[1], Rachmawati Wangsaputra[1], Abdul Hakim Halim[1]

[1] *Faculty of Industrial Technology, Institut Teknologi Bandung (ITB), Jl. Ganesha no 10, Bandung, 40132, Indonesia*

Abstract. This paper addresses a batch scheduling problem for a three-stage hybrid flowshop consisting of a machining stage processing common and unique components on unrelated parallel machines, an assembly stage combining the components into assembled products with complex assembly structures, and a differentiation stage processing the assembled products on dedicated machines to produce different product types. The common components are the same for all products and are processed in batches, while the unique components are dedicated to respective given product types and are processed individually (one-by-one component). The goal is to schedule all the products with different assembly structures to minimize total actual flow time (*TAFT*) defined as total time interval of components to be processed from their arrival times to their common due date. A non-linear programming model is proposed, where small size problems can be solved optimally using the LINGO software, and large size problems is to be solved using a heuristic algorithm. The proposed algorithm consists of two sub-algorithms. The first one is constructed using a shortest processing time (SPT) based heuristic to get a job sequence as an initial solution and the second one is to improve the initial solution using the variable neighbourhood descent (VND) method with neighbourhood insert and swap move operators. In solving the problem with the algorithm, two scenarios arise, e.g., the same and the different sequences for all stages. A set of hypothetical data is generated for different hierarchical assembly structures to test the model and the algorithm, and the results show that the different sequences for all stages obtain solutions with better performances than the same ones.

Keywords: Batch scheduling; Hierarchical assembly structures; Three-stage hybrid flowshop; Total actual flow time; Unique and common component

1. Introduction

This paper deals with a three-stage hybrid flowshop consisting of machining, assembly, and differentiation stages. The following pieces of research also describe the three-stage flowshops in different perspectives. Babayan and He (2004) and Utama et al. (2019) have studied the stages consisting of all machining processes, Maleki-Darounkolaei et al. (2012) and Komaki et al. (2017) have considered the three-stage flowshop as machining, transportation, and assembly stages, Futatsuishi et al. (2002) have dealt with the flowshop where the stages consist of assembly, differentiation, and packaging stages, and Xiong et al. (2015) have addressed the machining, assembly, and differentiation stages. In the mentioned studies, the systems adopt a forward job scheduling approach and consider only a single assembly operation, but there are practical cases where the approach adopted

[*] Corresponding author's email: rahmimaulidya@gmail.com, Tel.:+62-22-2504189, Fax. +62-22-2509164

doi: 10.14716/ijtech.v11i3.3555

should be a backward scheduling and the assembly is with complex structures. In addition, it can be noted that all the above papers deal only with job processing at the machining stage; however, in practical situations, there are cases where the process can be conducted in batches for components with the same type. This paper considers a backward scheduling approach for a three-stage hybrid flowshop consisting of machining, assembly, and differentiation stages where the assembly is with complex structures and the process can be conducted in batches.

Hwang et al. (2014) explains that the common component is one that can be found in all product types, and can be processed in batches sharing the same setup time, while the unique component is one with a one-by-one processes as dedicated to a given product type. The time when the components are assembled is synchronized according to a particular hierarchical assembly structure of the products. Komaki et al. (2018) point out that an assembly structure is considered complex when it has more than one assembly operation in at least one assembly level. Maulidya et al. (2018) have developed research for the case of unique and common components but ignoring the assembly structure of products. It can be observed that the practical situation of platen and assembly roller (spare parts of a printer) production applies different assembly structures processed in the same shop floor, and there is a CNC machine that can process both a unique shaft and a common bushing for platen types, and then the first assembly operation can be conducted for both components. This is the reason for this research to be conducted.

Furthermore, Cheng et al. (2009) and Xiong et al. (2015) have discussed a final process, one that occurs after the assembly stage, for producing different product types conducted at the so-called differentiation stage. The output products can be different in colors as a result of the painting machines (Xiong et al., 2015), and they can also be different in sizes as a result of the cutting machines (Lin and Liao, 2003). These studies applied job processing but in practical situations, there are cases where a differentiation stage is conducted in batches because each dedicated machine only processes the same product type. Huang and Lin (2013) have applied a batch scheduling problem at the differentiation stage, but this is only for a two-stage differentiation flowshop.

This research aims to develop a scheduling model considering production of unique and common components with a complex assembly structure of products and apply batch scheduling to the differentiation stage. This paper uses the objective of minimizing total actual flow time (TAFT) instead of the traditional criteria, let say, completion time (see Thawongklang and Tanwanichkul, 2016), flow time (see Xiong et al., 2015), and makespan (see Natesan and Chokkalingam, 2019). The total actual flow time is defined as the time interval from the arrival times of components to be processed to their product common due date, and this objective ensures that all finished product can be completed at their due date (Halim et al., 1994b). This paper proposes a non-linear programming model for a three-stage flowshop processing unique and common components with hierarchical assembly structures and an algorithm to solve the model. The proposed algorithm adopts the VND method to improve an initial solution obtained by applying the SPT rule. The VND is a metaheuristic for solving combinatorial and global optimization problems where the neighborhoods are searched in a systematic manner, known as a steepest descent heuristic to get better solutions (Vanchipura et al., 2014). This paper is organized as follows: Section 2 discusses the model formulation, Section 3 discusses the proposed algorithm, Section 4 discusses the numerical experiences, and Section 5 discusses the conclusion.

2. Methods

2.1. Problem Definition

There are n different jobs to be processed in three stages, namely machining, assembly, and differentiationstages. All jobs belong to a disjoint set of jobs of $N_1 = \{J_1, J_2, \ldots, J_{N_1}\}$ to $N_h = \{J_{N_1+1}, J_{N_1+2}, \ldots, J_{N_h}\}$. Each job J_i (Figure 1) has k common ($J_{i,k}$) and k unique components ($J_{i,k}$) to be processed on k different machines (unrelated) in parallel $M_{1,k}$ at the machining stage. A pair of a unique and a common component from a machine $M_{1,k}$ is assembled following a precedence network of the product (Figure 2) that represents the hierarchical assembly structure from the lowest Level o of the sub-assembled parts to Level 1 as the final assembly level for a product. The assembly stage consists of m_2 levels, each with a machines, $M_{2,o,l}$. After a job is assembled, the final process is held on machine $M_{3,h}$ at the differentiation stage where J_i belongs to type h. Figure 3 shows the system represented. In this paper, a model formulation and solution method are discussed to schedule all the jobs where the unique components are processed individually, and the common components are processed in batches with a setup time for each batch. The assembly operation of the jobs is conducted based on a precedence network. Finally, jobs are grouped into batches to be processed on the dedicated machines. The assumptions are as follows: all processing jobs are finished at their common due date, each machine can process at most one job at a time, and each job can be processed on at most one machine at a time. Furthermore, each batch is only for jobs of the same product type, and setup is conducted before the batch arrives at the machine. In addition, there are no pre-emptions allowed when a batch and a job are processed, and any additional processes between the assembly stage are neglected.

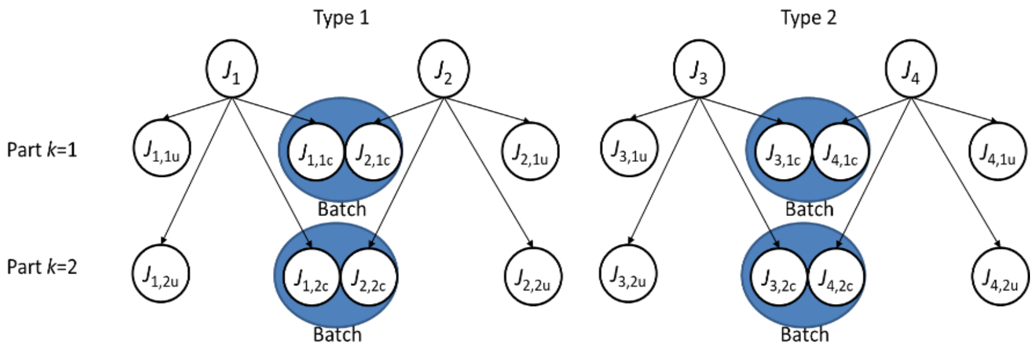


Figure 1 The processing of components at the machining stage

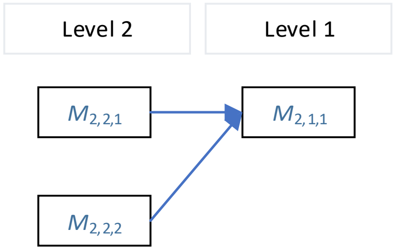


Figure 2 Assembly operation

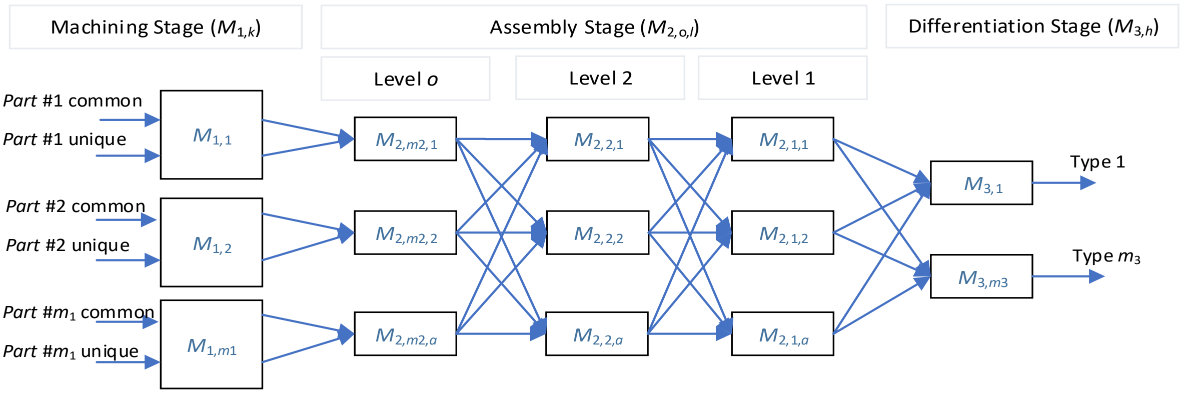


Figure 3 The three-stage hybrid flowshop system

2.2. Formulation

The notation in this paper is as follows.

Indices

j : index of jobs, *j*=1, 2…*n*

i : index of job positions, *i*=1, 2…*n*

u : index of batches, *u*=1, 2…*v*

c : index of position for unique jobs and common batches position, *c*=1,2…*γ*

m : index of stages, the machining (m_1), the assembly (m_2) and the differentiation (m_3) stages

k : index of machines at the machining stage processing common and unique components, *k*=1.., m_1

o,o': index of level operation $M_{2,o,l}$ at the assembly stage, *o*=1.., m_2

l,l' : index of machines $M_{2,o,l}$ at the assembly stage, *l*=1.., *a*

h : index of machines in the differentiation stage ($M_{3,h}$) that refers to a product type, *h*=1.., m_3

Parameters

n : number of jobs to be processed

v_m : number of batches for stage *m*

d : common due date

s_h : setup time for processing type *h* jobs

$p^{(1u)}_{j,k}$: the processing time of unique component in the machining stage

$p^{(1c)}_{j,k}$: the processing time of common component in the machining stage

$p^{(2)}_{j,o,l}$: the processing time of assembly in machine $M_{2,o,l}$ in the assembly stage

$p^{(3)}_{j,h}$: the processing time in machine $M_{3,h}$ at the differentiation stage

Φ : a large number

$r^2_{j,o',l',o,l}$: precedence network of job *j* from level o' and machine l' to level *o* and machine *l*

Decision variables

$X^{(m)}_{j[i_m]}$: binary variable to assign job *j* to position i_m at stage *m*

$W^{(m)}_{j[u_m]}$: binary variable to assign job *j* to batch u_m at stage *m*

$Y^{(m)}_{[u_m],h}$: binary variable to assign batch u_m to type *h* at stage *m*

$Q^{(m)}_{[u_m]}$: batch size u_m at stage *m*

$UP_{[c][i_1]}$: binary variable to assign position i_1 to new position *c* at stage *m*

$CP_{[c][u_1]}$: binary variable to assign batch u_1 to new position *c* at stage *m*

$B^{(1)}_{[i_1],k}, B^{(1)}_{[u_1],k}, B^{(2)}_{[i_2],o,l}, B^{(3)}_{[u_3],h}$: starting time for certain conditions

The complete formulation for the model is as follows:

$$\text{Min } TAFT = \sum_{i_1=1}^{n}\left(d - \min_{k\le m_1}\left(B_{[i_1],k}^{(1)}\right)\right) + \sum_{u_1=1}^{v_1}\left(d - \min_{k\le m_1}\left(B_{[u_1],k}^{(1)}\right)\right).\, Q_{[u_1]}^{(1)}; \qquad \forall i_1, u_1, k. \qquad (1)$$

Subject to:

$$B_{[u_3],h}^{(3)} = Y_{[u_3],h}^{(3)}\left(d - \sum_{z_3=1}^{u_3}\left(Y_{[z_3],h}^{(3)}\sum_{j=1}^{n}\left(W_{j[z_3]}^{(3)}.\, p_{j,h}^{(3)}\right)\right) - \left(\sum_{z_3=1}^{u_3} Y_{[z_3],h}^{(3)} - 1\right) S_h\right);$$
$$z_3 \le u_3; \forall u_3, h \qquad (2)$$

$$B_{[i_2],1,l}^{(2)} + \sum_{j=1}^{n}\left(X_{j[i_2]}^{(2)}.\, p_{j,1,l}^{(2)}\right) \le \sum_{h=1}^{m_3}\left(B_{[u_3],h}^{(3)}\sum_{j=1}^{n}\left(X_{j[i_2]}^{(2)}.\, W_{j[u_3]}^{(3)}\right)\right) + \left(1 - \sum_{j=1}^{n}\left(X_{j[i_2]}^{(2)}.\, W_{j[u_3]}^{(3)}.\, A_{j,1,l}^{(2)}\right)\right).\, \Phi; \forall l, i_2, u_3 \qquad (3)$$

$$B_{[i_2],o',l'}^{(2)} + \sum_{j=1}^{n}\left(X_{j[i_2]}^{(2)}.\, p_{j,o',l'}^{(2)}\right) \le B_{[i_2],o,l}^{(2)} + \left(1 - \sum_{j=1}^{n}\left(X_{j[i_2]}^{(2)}.\, r_{j,o',l',o,l}^{2}\right)\right)\Phi; \qquad o' > o; \forall l, i_2, o \qquad (4)$$

$$B_{[z_2],o,l}^{(2)} + \sum_{j=1}^{n}\left(X_{j[z_2]}^{(2)}.\, p_{j,o,l}^{(2)}\right) \le B_{[i_2],o,l}^{(2)} + \left(2 - \sum_{j=1}^{n}\left(X_{j[z_2]}^{(2)}.\, A_{j,o,l}^{(2)}\right) - \sum_{j=1}^{n}\left(X_{j[i_2]}^{(2)}.\, A_{j,o,l}^{(2)}\right)\right)\Phi;$$
$$i_2 < n, z_2 > i_2, \forall i_2, o, l \qquad (5)$$

$$B_{[i_1],k}^{(1)} + \sum_{j=1}^{n}\left(X_{j[i_1]}^{(1)}.\, p_{j,k}^{(1u)}\right) \le \left(B_{[i_2],o,k}^{(2)}.\sum_{j=1}^{n}\left(X_{j[i_1]}^{(1)}.\, X_{j[i_2]}^{(2)}\right)\right) + \Phi\left(1 - \sum_{j=1}^{n}\left(X_{j[i_1]}^{(1)}.\, X_{j[i_2]}^{(2)}.\, A_{j,o,l}^{(2)}\right)\right); \forall i_1, k, o, l \qquad (6)$$

$$B_{[u_1],k}^{(1)} + \sum_{j=1}^{n}\left(W_{j[u_1]}^{(1)}.\, p_{j,k}^{(1c)}\right) \le \left(B_{[i_2],o,k}^{(2)}.\sum_{j=1}^{n}\left(W_{j[u_1]}^{(1)}.\, X_{j[i_2]}^{(2)}\right)\right) + \left(1 - \left(\sum_{j=1}^{n}\left(W_{j[u_1]}^{(1)}.\, X_{j[i_2]}^{(2)}.\, A_{j,o,l}^{(2)}\right)\right)\right)\Phi; \forall i_2, u_1, k, o, l \qquad (7)$$

$$\sum_{i_1=1}^{n}\left(UP_{[c+1],[i_1]}.\left(B_{[i_1],k}^{(1)} + \sum_{j=1}^{n}\left(X_{j[i_1]}^{(1)}.\, p_{j,k}^{(1u)}\right)\right)\right) + \sum_{u_1=1}^{v_1}\left(CP_{[c+1],[u_1]}.\left(B_{[u_1],k}^{(1)} + \sum_{j=1}^{n}\left(W_{j[u_1]}^{(1)}.\, p_{j,k}^{(1c)}\right)\right)\right) \le \sum_{i_1=1}^{n}\left(UP_{[c],[i_1]}.\left(B_{[i_1],k}^{(1)}\right)\right) + \sum_{u_1=1}^{v_1}\left(CP_{[c],[u_1]}.\left(B_{[u_1],k}^{(1)} - \sum_{h=1}^{m_3} Y_{[u_1],h}^{(1)}.\, S_h\right)\right) \qquad c < \gamma;\ \forall k \qquad (8)$$

$$A_{j,h} = \begin{cases} 1, & \text{if } p_{j,h}^{(3)} > 0 \\ 0, & \text{if } p_{j,h}^{(3)} = 0 \end{cases} \qquad \forall j, h \qquad (9)$$

$$A_{j,o,l}^{(2)} = \begin{cases} 1, & \text{if } p_{j,o,l}^{(2)} > 0 \\ 0, & \text{if } p_{j,o,l}^{(2)} = 0 \end{cases} \qquad \forall j, o, l \qquad (10)$$

$$\sum_{j=1}^{n} X_{j[i_m]}^{(m)} = 1; \qquad m = 1,2; \forall i_m \qquad (11)$$

$$\sum_{i_m=1}^{n} X_{j[i_m]}^{(m)} = 1; \qquad m = 1,2; \forall j \qquad (12)$$

$$\sum_{c=1}^{\gamma} UP_{[c][i_1]} = 1; \qquad \forall i_1, c \qquad (13)$$

$$\sum_{c=1}^{\gamma} CP_{[c][u_1]} = 1; \qquad \forall u_1, c \qquad (14)$$

$$\sum_{i_1=1}^{n} UP_{[c][i_1]} + \sum_{u_1=1}^{v} CP_{[c][u_1]} = 1; \qquad \forall c \qquad (15)$$

$$\sum_{h=1}^{m_3}\sum_{u_m=1}^{\gamma_m}\left(Q_{[u_m]}^{(m)}.\, Y_{[u_m],h}^{(m)}\right) = n; \qquad m = 1,3; \forall h \qquad (16)$$

$$\sum_{u_m=1}^{\gamma_m}\left(W_{j[u_m]}^{(m)}.\, Y_{[u_m],h}^{(m)}\right) = A_{j,h}; \qquad m = 1,3; \forall j, h \qquad (17)$$

$$\sum_{j=1}^{n} W_{j[u_m]}^{(m)} = Q_{[u_m]}^{(m)}; \qquad m = 1,3; \forall u_m \qquad (18)$$

$$\sum_{h=1}^{m_3} Y_{[u_m],h}^{(m)} = 1; \qquad m = 1,3; \forall u_m \qquad (19)$$

$$\sum_{u_m=1}^{\gamma_m} W_{j[u_m]}^{(m)} = 1; \qquad m = 1,3; \forall j \qquad (20)$$

$$Q_{[u_m]}^{(m)} > 0; \qquad m = 1,3; \forall u_m \qquad (21)$$

$$X_{j[i_m]}^{(m)}, UP_{[c][i_1]}, CP_{[c][u_1]}, W_{j[u_m]}^{(m)}, Y_{[u_m],h}^{(m)} \in \{0,1\} \qquad (22)$$

$$B_{[i_1],k}^{(1)}, B_{[u_1],k}^{(1)}, B_{[i_2],o,l}^{(2)}, B_{[u_3],h}^{(3)} \geq 0 \qquad (23)$$

The objective function in Equation 1 is to minimize the total actual flow time that consists of the actual flow time of jobs of unique components and the actual flow time of batches of common components. Equation 2 explains the starting time of batch u_3 at the differentiation stage assigned from the due date. Equation 3 defines the starting time of the assembly stage level 1 from the differentiation stage. Equation 4 defines the starting time of the assembly stage level o using the precedence $r_{j,o',l',o,l}^{2}$. Equation 5 ensures that there is no overlapping in processing the jobs. Equations 6 and 7 show the connection between the lower level of the assembly stage and the job or the batch in the machining stage. Equation 8 ensures that the processing of jobs and batches are arranged by position *c.* Equation 9 is the binary variable that shows job *j* is processed on machine *h*. Equation 10 describes the binary variable for the precedence network defined from the processing time at the assembly stage. Equations 11 and 12 explain that a job can only be processed on one machine, and a machine can process only one job. Equations 13 to 15 arrange the position of job and batch into position *c.* Equations 16 to 18 describe the material balance for the batch. Equations 19 and 20 describe that a job only belongs to a batch, and the batch is for a specific product type. Equation 21 ensures a batch cannot be zero.

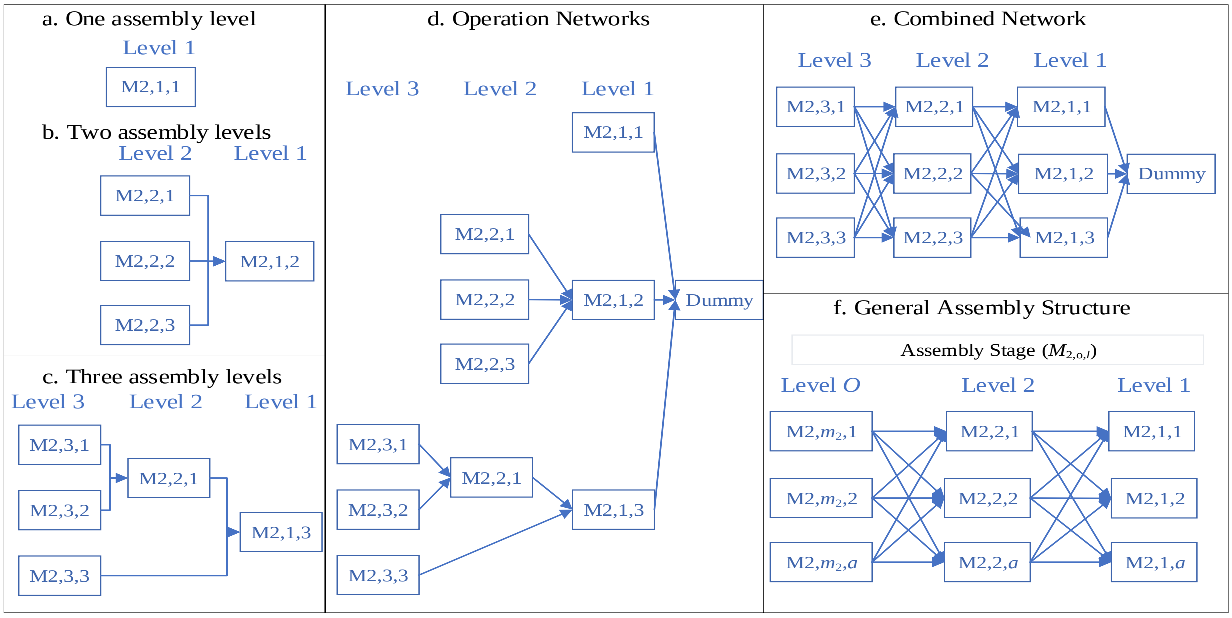


Figure 4 The alternative assembly levels for $m_1 = 3$ (a, b, c); the combining network (d); a representation of the combined network (e); and the general assembly structure (f)

The objective of total actual flow time for processing unique and common components has been formulated by Maulidya et al. (2018). The assembly stage consists of assembly levels built from the possible alternatives of assembling the components, such as a single assembly operation on $M_{2,1,1}$ (Figure 4a), a two-level assembly operation (Figure 4b), and a three-level assembly operation (Figure 4c). These assembly levels are generated from the number of machines in the machining stage, $m_1 = 3$. In generating an assembly structure, all assembly operations are combined into a network with a dummy as an end node (Figure 4d), then the operations are merged with the same machine (Figure 4e). This results in a general assembly structure for the assembly stage without using any dummy (Figure 4f).

A precedence network of $r^{(2)}_{j,o',l',o,l}$ is generated to represent the assembly structure illustrated in Figure 2 for $m_2 = 2$ and $a = 2$. Let o' and l' be the respective successors for the assembly level and the assembly machine of each level. For $j = 1$ (Table 1), the network of $M_{2,2,1} \rightarrow M_{2,1,1}$ is shown in $o' = 2$, $l' = 1 \rightarrow o = 1$, $l = 1$ and the network of $M_{2,2,2} \rightarrow M_{2,1,1}$ is shown in $o' = 2$, $l' = 2 \rightarrow o = 1$, $l = 1$. The $r^{(2)}_{j,o',l',o,l}$ for each network is equal to 1 to represent the matrix relation between o', l' and o, l.

Table 1 The precedence network for the first job (j=1) to define $r^{(2)}_{j,o',l',o,l}$

j	o'	l'	o	l	$r^{(2)}_{1,o',l',o,l}$	j	o'	l'	o	l	$r^{(2)}_{1,o',l',o,l}$	j	o'	l'	o	l	$r^{(2)}_{1,o',l',o,l}$	j	o'	l'	o	l	$r^{(2)}_{1,o',l',o,l}$
1	1	1	1	1	0	1	1	2	1	1	0	1	2	1	1	1	1	1	2	2	1	1	1
1	1	1	1	2	0	1	1	2	1	2	0	1	2	1	1	2	0	1	2	2	1	2	0
1	1	1	2	1	0	1	1	2	2	1	0	1	2	1	2	1	0	1	2	2	2	1	0
1	1	1	2	2	0	1	1	2	2	2	0	1	2	1	2	2	0	1	2	2	2	2	0

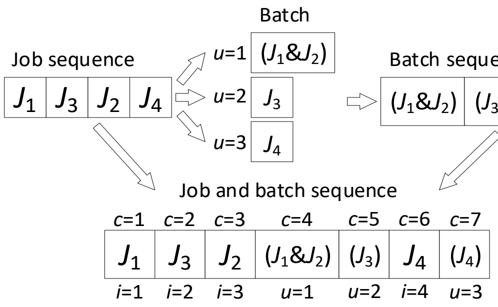
2.3. Algorithm

The proposed algorithm begins with a construct algorithm using a SPT-based heuristic to generate job sequences for an initial solution. Inspired by the benefits of the SPT-based heuristic for the backward approach in Halim et al. (1994a), a SPT-based heuristic is proposed for the represented system. The procedure of the SPT-based heuristic is as follows:

Step 1: Generate job sequences in increasing order of the processing times of each stage (different sequences scenario): the first stage is taken by the maximum processing time of unique component, common component, and all components for each job, the second stage is taken by the maximum processing time of all levels for each job, and the third stage by the maximum processing time for each job. The same sequence scenario for all stages is taken by the cumulative processing time of the first and the second stages, the second and the third stages, and then all three stages, respectively.

Step 2: Calculate TAFT value and select the job sequence using the minimum TAFT.

The decoding process is held whenever there is a job sequence. The decoding (Figure 5a) consists of generating a batch sequence and a job-and-batch sequence. To generate a batch sequence, the number of batches (v) is determined and the batch size ($Q_{[u]}$) is calculated following the study by Halim et al. (1994a). In generating a batch sequence, a set of jobs is placed in batches, respectively; the first job will set the batch type and the next job is placed after the first job only if the product type is matched, and the batch sizes are not exceeded. If a job does not match a batch type, then that job moves to another batch. After all the batches are set, then the batch sequence is obtained. Next, the illustration in Figure 5a generates a job and batch sequence that represents the process of unique and common components. The input is the job sequence and the batch sequence. The rules for generating the sequence are: the unique component (a job) is scheduled first followed by the common components (a batch), and a batch is placed in the sequence after all members similar to jobs are assigned in the sequence. The proposed scenarios are the same sequence for all stages and the different sequences for each stage. Furthermore, a VND procedure (Figure 5b) is proposed to improve the initial solution in a deterministic way using an operator of the insert move (k = 1) and swap move (k = 2). The insert move is a process of inserting a selected position into another possible position along the solution *x*. The swap move is a process of exchanging two positions in the solution *x*.


Batch
Job sequence
J_1 J_3 J_2 J_4
u=1 (J_1&J_2)
u=2 J_3
u=3 J_4
Batch sequence
(J_1&J_2) (J_3) (J_4)
Job and batch sequence
c=1 c=2 c=3 c=4 c=5 c=6 c=7
J_1 J_3 J_2 (J_1&J_2) (J_3) J_4 (J_4)
i=1 i=2 i=3 u=1 u=2 i=4 u=3

(a)

```
**The procedure of VND**
$x^o \leftarrow initial\ solution$
Decoding
$x' \leftarrow x^o$
$f(x') \leftarrow f(x^o)$
Do
    $k \leftarrow 1$
    $x'' \leftarrow operator(k, x')$
    Decoding
    If $f(x'') < f(x')$
        $x' \leftarrow f(x'')$
        $k \leftarrow 1$
    else
        $k \leftarrow k + 1$
While $k \leq k_{max}$
$x^* \leftarrow x'$
$f(x^*) \leftarrow f(x')$
```

(b)

Figure 5 (a) Illustration for the decoding process; and (b) The procedure of VND

3. Results and Discussion

The numerical experiences are conducted using a set of parameters, such as the number of the jobs processed (n = 4, 6), the number of batches (v_1 = 2, 3; v_3 = 2, 3), the number of machines at each stage (m_1= 2, 3; a = 2, 3; m_2= 2, 3; m_3= 2, 3), the precedence network (one, two and three assembly levels), and the setup times for type 1 and type 2 are respectively 3 and 5. The illustration for n = 4 follows the product structure (Figure 1), the assembly structure (Figure 2), the processing time of jobs (Table 2) and are run with *Processor Intel(R) Xeon(R) CPU E5-1620 v2 @3.70GHz 3.70GHz RAM 8.00.* The proposed algorithm is set to the same sequence for all the stages (see Table 3 for the SPT result) and to the different sequence in each stage (see Table 4 for the SPT result). To optimize the initial solution, a VND procedure is conducted.

Table 2 The processing time data for n = 4

Job	Type	$p_{j,1}^{(1c)}$	$p_{j,1}^{(1u)}$	$p_{j,2}^{(1c)}$	$p_{j,2}^{(1u)}$	$p_{j,1,1}^{(2)}$	$p_{j,1,2}^{(2)}$	$p_{j,2,1}^{(2)}$	$p_{j,2,2}^{(2)}$	$p_{j,1}^{(3)}$	$p_{j,2}^{(3)}$
J_1	1	3	5	2	4	4	0	4	5	6	0
J_2	1	3	6	4	3	9	0	9	4	10	0
J_3	2	3	6	1	4	3	0	6	3	0	9
J_4	2	4	3	5	5	9	0	5	9	0	5

Table 3 The result of SPT for the same sequence scenario ($v_1 = v_3 = 3$)

S_x	The same sequence	1st stage	2nd stage	3rd stage	*TAFT*
S_1	J_1-J_3-J_4-J_2 (J_1&J_2)-(J_3)-(J_4) J_1-J_3-J_4-J_2-(J_1&J_2)-(J_3)-(J_4)	J_1-J_3-J_4-J_2-(J_1&J_2)-(J_3)-(J_4)	J_1-J_3-J_4-J_2	(J_1&J_2)-(J_3)-(J_4)	436*
S_2	J_1-J_4-J_3-J_2 (J_1&J_2)-(J_4)-(J_3) J_1-J_4-J_3-J_2-(J_1&J_2)-(J_4)-(J_3)	J_1-J_4-J_3-J_2-(J_1&J_2)-(J_4)-(J_3)	J_1-J_4-J_3-J_2	(J_1&J_2)-(J_4)-(J_3)	445
S_3	J_1-J_4-J_3-J_2 (J_1&J_2)-(J_4)-(J_3) J_1-J_4-J_3-J_2-(J_1&J_2)-(J_4)-(J_3)	J_1-J_4-J_3-J_2-(J_1&J_2)-(J_4)-(J_3)	J_1-J_4-J_3-J_2	(J_1&J_2)-(J_4)-(J_3)	445

Table 4 The result of SPT for different sequence scenarios

S_x	1st stage $Q^{(1)}_{[u_1]} = [2\ 1\ 1]$	2nd stage	3rd stage $Q^{(1)}_{[u_1]} = [2\ 1\ 1]$	*TAFT*
S_1	J_1-J_3-J_2-J_4 (J_1&J_2)-(J_3)-(J_4) J_1-J_3-J_2-(J_1&J_2)-(J_3)-J_4-(J_4)	J_1-J_3-J_2-J_4	J_4-J_1-J_3-J_2 (J_4&J_3)-(J_1)-(J_2)	374*
S_2	J_1-J_4-J_2-J_3 (J_1&J_2)-(J_4)-(J_3) J_1-J_4-J_2-(J_1&J_2)-(J_4)-J_3-(J_3)	J_1-J_3-J_2-J_4	J_4-J_1-J_3-J_2 (J_4&J_3)-(J_1)-(J_2)	460
S_3	J_1-J_4-J_2-J_3 (J_1&J_2)-(J_4)-(J_3) J_1-J_4-J_2-(J_1&J_2)-(J_4)-J_3-(J_3)	J_1-J_3-J_2-J_4	J_4-J_1-J_3-J_2 (J_4&J_3)-(J_1)-(J_2)	460

The optimal model can still give the global solution from the small instance (n = 4) in less than ten hours, while the other instance (n = 6) only shows the local optimal for about ten hours. The optimal model shows that the result for the different sequence is better than the result for the same sequence. The proposed algorithm has been tested to observe the impact of the different and the same sequences on the system. Table 5 shows the results for the optimal solution and the approaches for each stage with its TAFT. The optimal schedule can be seen in Figure 6.

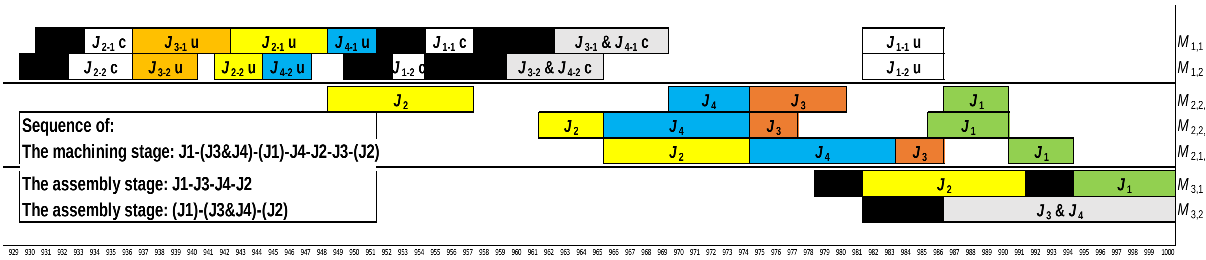


Figure 6 Gantt chart of the optimal schedule using *Lingo16* (*TAFT*=332)

The optimal model applies the different sequences for each stage, and the results show the optimal solution, except for Set 1 in Table 5, where the same sequence shows the best TAFT. In the proposed algorithm, the rule for generating the job-batch sequence should be evaluated in order to improve the solution. The experiments for different assembly levels show that the model and its algorithm can perform on one, two, and three assembly levels with a variation on the number of machines at the machining stage and the differentiation stage. The trial for different assembly levels need to be explored, and this leaves plenty of room for further research on this topic.

Table 5 The result comparison of *TAFT* between the optimal solution and the solution from the proposed algorithms (the same and the different sequences scenarios)

Data	n	m_1	m_2	a	m_3	v_1	v_3	*TAFT* Optimal	*TAFT* Same seq	*TAFT* Different seq
Set 1	4	2	2	2	2	2	2	396	393	409
Set 2	4	2	2	2	2	2	3	347	386	352
Set 3	4	2	2	2	2	3	2	384	489	468
Set 4	4	2	2	2	2	3	3	332	392	374
Set 5	4	2	2	2	3	3	3	327	375	356
Set 6	4	3	3	3	2	2	2	445	527	489
Set 7	4	3	3	3	2	2	3	420	488	462
Set 8	4	3	3	3	2	3	2	648	754	694
Set 9	4	3	3	3	2	3	3	393	450	434
Set 10	4	3	3	3	3	3	3	398	512	476
Set 11	4	3	2	3	2	2	2	331	436	408
Set 12	4	3	2	3	2	2	3	358	432	386
Set 13	4	3	2	3	2	3	2	467	498	486
Set 14	4	3	2	3	2	3	3	338	423	387
Set 15	4	3	2	3	3	3	3	393	532	498

4. Conclusions

This paper addresses the three-stage hybrid flowshop for processing different product types considering hierarchical assembly structures to minimize TAFT. Based on the numerical experiences, it is possible to use both scenarios, but the different sequence scenario provides the minimum *TAFT* value. This experiment is also conducted to evaluate the precedence network, of which the model is effective in providing the solution. For further research, the condition can be extended into multiple due dates to reflect the practice where the finished product is delivered in different due dates, and pre-emption is allowed.

Acknowledgements

The authors thank the Ministry of Research, Technology and Higher Education for the dissertation research funding under contract No.011/KM/PNT/2018 with Universitas Trisakti.

References

Babayan, A., He, D., 2004. Solving the n-job 3-stage Flexible Flowshop Scheduling Problem using an Agent-based Approach. *International Journal of Production Research*, Volume 42(4), pp. 777–799

Cheng, T.C.E., Lin, B.M.T., Tian, Y., 2009. Scheduling of a Two-stage Differentiation Flowshop to Minimize Weighted Sum of Machine Completion Times. *Computers & Operations Research*, Volume 36(11), pp. 3031–3040

Futatsuishi, Y., Watanabe, I., Nakanishi, T., 2002. A Study of The Multi-stage Flowshop Scheduling Problem with Alternative Operation Assignments. *Mathematics and Computers in Simulation*, Volume 59(1), pp. 73–79

Halim, A.H., Miyazaki, S., Ohta, H., 1994a. Batch Scheduling Problems to Minimize Actual Flow Times of Parts through the Shop under JIT Environment. *European Journal of Operational Research*, Volume 72(3), pp. 529–544

Halim, A.H., Miyazaki, S., Ohta, H., 1994b. Lot Scheduling Problems of Multiple Items in the Shop with Both Receiving and Delivery Just in Times. *Production Planning & Control: The Management of Operations*, Volume 5(2), pp. 175–184

Huang, T.C., Lin, B.M.T., 2013. Batch Scheduling in Differentiation Flowshops for Makespan Minimization. *International Journal of Production Research*, Volume 51(17), pp. 5073–5082

Hwang, F.J., Kovalyov, M.Y., Lin, B.M.T., 2014. Scheduling for Fabrication and Assembly in a Two-machine Flowshop with a Fixed Job Sequence. *Annals of Operations Research*, Volume 217(1), pp. 263–279

Komaki, G.M., Teymourian, E., Kayvanfar, V., Booyavi, Z., 2017. Improved Discrete Cuckoo Optimization Algorithm for the Three-stage Assembly Flowshop Scheduling Problem. *Computers & Industrial Engineering,* Volume 105, pp. 158–173

Komaki, G.M., Sheikh, S., Malakooti, B., 2018. Flow Shop Scheduling Problems with Assembly Operations: A Review and New Trends. *International Journal of Production Research*, Volume 57(10), pp. 2926–2955

Lin, H.T., Liao, C.J., 2003. A Case Study in a Two-stage Hybrid Flowshop with Setup Time and Dedicated Machines. *International Journal of Production Economics*, Volume 86(2), pp. 133–143

Maulidya, R., Suprayogi, Wangsaputra, R., Halim, A.H., 2018. Batch Scheduling for Unique and Common Components in a Hybrid Assembly Differentiation Flow Shop to Minimize Total Actual Flow Time. *In:* Conference Paper of 19th Asia Pacific Industrial Engineering & Management System (APIEMS) Conference 2018, Dec 2-5, Hong Kong

Maleki-Darounkolaei, A., Modiri, M., Tavakkoli-Moghaddam, R., Seyyedi, I., 2012. A Three-Stage Assembly Flow Shop Scheduling Problem with Blocking and Sequence-dependent Set Up Times. *Journal of Industrial Engineering International*, Volume 8(1), pp. 1–7

Natesan, G., Chokkalingam, A., 2019. Optimal Task Scheduling in the Cloud Environment using a Mean Grey Wolf Optimization Algorithm. *International Journal of Technology*, Volume 10(1), pp. 126–136

Thawongklang, K., Tanwanichkul, L., 2016. Application of Production Scheduling Techniques for Dispatching Ready-Mixed Concrete. *International Journal of Technology*, Volume 7(7), pp. 1163–1170

Utama, D.M., Widodo, D.S., Wicaksono, W., Ardiansyah, L.R., 2019. A New Hybrid Metaheuristics Algorithm for Minimizing Energy Consumption in the Flow Shop Scheduling Problem. *International Journal of Technology*, Volume 10(2), pp. 320–331

Vanchipura, R., Sridharan, R., Babu, A.S., 2014. Improvement of Constructive Heuristics using Variable Neighbourhood Descent for Scheduling a Flow Shop with Sequence Dependent Setup Time. *Journal of Manufacturing Systems*, Volume 33(1), pp. 65–75

Xiong, F., Xing, K., Wang, F., 2015. Scheduling a Hybrid Assembly-Differentiation Flow Shop to Minimize Total Flow Time. *European Journal of Operation Research*, Volume 240, pp. 338–354