Journal of Optimization in Industrial Engineering Vol.12, Issue 1, Winter and Spring 2019, 63- 78
DOI: 10.22094/joie.2018.760.1484

Modeling and Solving the Job Shop Scheduling Problem Followed by an Assembly Stage Considering Maintenance Operations and Access Restrictions to Machines

Seyed Mohamad Hassan Hosseini

Department of Industrial Engineering and Management, Shahrood University of Technology, Shahrood, Iran

Received 26 April 2017; Revised 11 November 2017; Accepted 25 April 2018

Abstract

This paper considers job shop scheduling problem followed by an assembly stage and Lot Streaming (LS). It is supposed here that a number of products have been ordered to be produced. Each product is assembled with a set of several parts. The production system includes two stages. The first stage is a job shop to produce parts. Each machine can process only one part at the same time. The second stage is an assembly shop that contains several parallel machines. Maintenance operations and access restrictions to machines in the first stage are also considered. The objective function is to minimize the completion time of all products (makespan). At first, this problem is described and modelled as a mixed integer linear programming, and GAMS software is applied to solve small-sized problems. Since this problem has been proved to be strongly NP-hard, two new algorithms based on GA and SA are developed to solve the medium- and large-sized problems. In order to verify the effectiveness of the proposed algorithms, a statistical analysis is used along with Relative Percentage Deviation (RPD) factor and well-known criterion. IMP. Various problems are solved by the proposed algorithms. Computational results reveal that both of the two proposed algorithms have good performance. However, the method based on the genetic algorithm performs better than the other proposed algorithm with respect to the objective function.

***Keywords*:** Job shop; Assembly; Maintenance operations; Access restrictions to machines.

1. Introduction

Scheduling is one of the most important activities in the production planning and control systems (Cummings and Egbelu, 1998). This is because of strong competition and limitation of resources in our environment (Maleki-Darounkolaei et al., 2012). The job shop scheduling problem (JSP) has been considered as a notoriously stubborn combinatorial optimization problem since the 1950s (Zhang and Cheng, 2011). In the job shop scheduling problem, a finite set of jobs is processed on a finite set of machines. Each job is characterized by a fixed order of operations, each of which is to be processed on a specific machine for a specified duration. Each machine can only process one job at the same time, and a job only initiates processing on a given machine. The aim of JSP is to find a schedule that optimizes a specified objective function such as makespan.

Usually, scheduling for the parts machining and planning for the assembly operations have been independently considered (Yokoyama and Santos, 2005); however, this may not lead to the best results for the total production system. Hence, the two-stage assembly scheduling problem that has many applications in industries and has received increasing attention of researchers lately. Lee et al. (1993) described an application in a fire engine assembly plant while Potts et al. (1995) described another application in personal computer manufacturing. In particular, manufacturing of almost all items may be modelled as a two-stage assembly scheduling (Allahverdi A, Al-Anzi F. S., 2009). Hence, after Lee et al. published their paper about 3-machine assembly-type flow shop scheduling problem in 1993, the assembly scheduling problems have received considerable attention from researchers during the last two decades.

In these production systems, usually, there is a machining stage and an assembly stage. Machining stage that can be a single machine, parallel machines, flow shop, hybrid flow shop or job shop processes and fabricates the parts (components) independently. Assembly stage performs assembly (joining) operations into product. The main criterion for this problem is the minimization of the maximum job completion time (makespan) (Koulamas Ch., Kyparisis G.J., 2001).

In this paper, a two-stage assembly scheduling problem is studied in which that the first stage is a job shop. In the classic format of job shop scheduling systems, a lot is usually invisible and the entire lot must be completed before being transferred to its successor machine (chen and steiner, 1997). Lot streaming (LS) technique was firstly introduced by Reiter (1966) as a methodology of splitting a job into a number of smaller sub-jobs, such that its successive operations can be overlapped in different stages. This increases the material flow between machines and, so, reduces the makespan. It is clear that, under a condition when more than one of each product is needed, determining the batch sizes must be considered. According to the studies around this problem, most of

*Corresponding author Email address: sh.hosseini@shahroodut.ac.ir

researchers have presented an iterative procedure to solve this problem. This procedure first computes the sub-lot sizes and, then, determines the sequence of sub-lots on the machines. Dauzere-Peres and Lasserre (1993) presented a model and an iterative procedure in a general job shop environment with lot steaming in order to improve the makespan. In their suggested procedure, at first, the optimal sub-lot sizes were computed, and then, a better sequence was computed by solving a standard job shop scheduling problem with fixed sub-lot sizes. Wagner and Ragatz (1994) studied the job shop scheduling problem with lot splitting to minimize the number of tardy jobs. They also considered the impact of setup times and the size of the transfer batches on lot splitting performance. Jeong et al. (1999) presented an iterative approach that minimizes the makespan of job shop scheduling problem with lot streaming. Their formulated problem reflected the real manufacturing environment by considering setup times and due dates. They proposed a modified shifting bottleneck procedure to solve the problem. Buscher and Shen (2009) focused on solving the lot streaming in a job shop environment. This research is one of the pioneer studies that tried to solve this kind of problems with meta-heuristic algorithms. They presented a three-phase algorithm which consists of the predetermination of sub-lot sizes, determination of schedules, and the variation of sub-lot sizes based on tabu search. Because an iterative procedure is a common method used in order to solve the job shop scheduling problem with lot streaming, they used an iterative procedure in the first phase and a hierarchical approach in the second and third phases. Lei and Guo (2013) considered lot-streaming problem in a job shop with consistent sub-lots and transportation. A modified artificial bee colony algorithm has been proposed by them to minimize makespan. Demir and Isleyen (2014) presented a flexible job shop scheduling problem with overlapping in operations. They developed a new mathematical model. In addition, a genetic algorithm with an effective chromosome representation and new decoding methodology was proposed by in their paper.

As mentioned, this problem has many applications in industry and, therefore, has received increasing attention from many researchers (Seyedi, Maleki-Daronkolaei and Kalashi, 2012; Fattahi, Hosseini and Jolai, 2013; Al-Anzi and Allahverdi, 2013; Navaei et al., 2014; Xiong, Xing and Wang, 2015). Chan et al. (2008) extended the application of lot streaming to an assembly job shop problem for the first time. In their model, the objective function minimizes inventory and lateness costs. In addition, they presented an efficient algorithm based on the genetic algorithm and simple dispatching rules. In another paper presented by Chan et al. (2009), the previous studies of LS to AJSP were extended by allowing part sharing among distinct products. The objective function of the proposed model is to minimize the lateness costs. In addition to the use of simple dispatching rules, they proposed an evolutionary approach with genetic algorithm to solve the problem. Wong et al. (2009) investigated a resource-constrained assembly job shop-scheduling problem with a lot streaming technique. The objective of their presented model is to minimize the total lateness cost of all final products. They also used common part ratio and workload index to enhance the model's usefulness. Since the complexity of AJSSP is NP-hard, an innovative approach with genetic algorithm was suggested in order to solve the problem. Wong and Ngan (2013) studied the assembly job shop scheduling with the lot steaming. In this study, their objective function is to minimize the makespan time. They considered part sharing ratio (PSR) and system congestion index (SCI) to differentiate product-specific components from general components and creating different starting conditions of the shop floor. In order to solve the problem, they proposed a hybrid genetic algorithm (HGA) and a hybrid particle swarm optimization (HPSO). Daneshamoz et al. (2013) proposed a mixed integer linear programming model for job shop scheduling with a parallel assembly stage in order to minimize makespan. In addition, they suggested a particle swarm optimization algorithm to solve the problem. Their results showed that suggested algorithm could reach near-optimal solutions in various dimensions of problem. Yao and Sarin (2014) addressed a lot steaming problem for a two-stage assembly system involving multiple lots with the objective of minimizing the makespan. They proposed a branch-and-bound-based methodology for this problem that relies on effective lower bounds and dominance properties. Mohammadi (2016) addressed a multi-objective job shop scheduling problem by considering an assembly stage and lot streaming. The objective function is to minimize the makespan and total weighted earliness and tardiness penalties. A mixed integer linear programming model was presented in this research. In addition, a meta-heuristic algorithm based on a Harmony Search was proposed to solve the problem. Nejati et al. (2016) considered a two-stage assembly hybrid flow shop scheduling problem with a work shift constraint and lot streaming to minimize the sum of weighted completion times of products in each shift. A mixed integer non-linear programming model was introduced to describe the problem. In addition, they used a genetic algorithm and simulated annealing to determine the best sequence and scheduling for the problem.

Another important fact is that, in most real-life industries, it is possible for machines to be unavailable due to unforeseen breakdowns or due to their need for scheduled PM activities, where the corresponding unavailability periods are known in advance (deterministic unavailability). Generally, PMs are planned for time intervals of the planning horizon to restore the reliability of a machine before it breaks down in order to maintain the equipment as well as the shop and provide better overall availability (Khoukhi et al., 2017). It should be noted that, in some systems, maintenance can only be done if all the production lines stop. In other management policies, maintenance and production are planned jointly. However, it is important to integrate production decisions into developing optimal PM policies (Mollahassani-pour, Abdollahi, & Rashidinejad, 2014; Xiao, Song, Chen, & Coit, 2016).

The aggregated problem of production scheduling and maintenance planning leads to higher efficiency because these two activities interrelate and both of them occupy the machine's capacity. In this respect, production depletes the machine's reliability and maintenance restores its reliability. Production scheduling and PM planning should be considered along with the integrated optimization model to balance the utilization and availability of the resource. Therefore, deterministic scheduling problems with maintenance and non-availability intervals have received considerable attention since the beginning of the 1990s (Wang & Liu, 2014, and Cui et al., 2018).To the best of our knowledge, according to the literature review, no one has considered a job shop scheduling problem with the lot streaming technique followed by an assembly stage considering the maintenance operations and access restriction to machines. Since the JSSP is known to be a NP-hard optimization problem (Garey, Johnson and sethi, 1976), this problem is more complex by adding a parallel assembly stage and the lot streaming technique to the classical job shop scheduling. Therefore, it is necessary to solve the medium- and large-sized instances with effective meta-heuristics to find the optimal or near-optimal solutions in a reasonable amount of time.

In order to structure a relevant literature review of the considered problem and to show the main contributions of this study, we have classified most previous papers according to three characteristics: definition of the problem, modeling and solution methods, and objectives (see Table 1). The characteristics of our paper are presented in the last row of Table 1.

The paper is organized as follows. In section 2, the problem is described in detail and the assumptions and mathematical model are presented. Section 3 describes the solution approach and proposed algorithms. Computational results are reported in section 4. Finally, conclusions and recommendations for future research are described in Section 5.

2. Problem Description

The considered problem in this study is a job shop scheduling problem with the lot streaming and a parallel assembly stage. This system can be defined as a problem consisting of assignment, sequencing, and determining lot sizes of jobs in a 2-stage production system. At the first stage, a set of J jobs $(J_1, J_2, \dots, J_{n_p})$ of P products $(P_1, P_2, \dots, P_p)$ is processed on a set of M machines $(M_1, M_2, \dots, M_m)$; then, they are stored in the inventory station until all lots from the BOM of the same product are completed and at least one unit of the final product can be produced. Each job of each product has $h_{j,p}$ operations and $s_{j,p}$ sub-lots must be processed with processing time $ps_{j,p,h}$. Then, the assembly operations starts at a parallel assembly stage for the final product assembly if at least one machine is unemployed at the second stage. Otherwise, they will be sent to the inventory station. Each sub-lot of each product must be assembled with assembly time $A_p$on a machine from a set of M'machines $(M'_1, M'_2, \dots, M'_m)$. Figure 1 shows the layout of the considered production system.

Basic assumptions of this problem are provided as below:

- ✓ There is no machine breakdown.
- ✓ No preemption of operations (sub-lots) is allowed.
- ✓ Each machine can process only one job (sub-lot) at the same time.
- ✓ Each job can be processed by only one machine at the same time.
- ✓ Each job has a fixed processing route which traverses all the machines in a predetermined order.
- ✓ Demand for the final products is specified and all jobs are available at time zero.
- ✓ A typical BOM of a product defines the assembly relationship between the root components (jobs or lots) in the system.
- ✓ Time processing of jobs and assembly of products is deterministic.
- ✓ When all lots from the BOM of the same product are completed in the work station, the assembly operations can start that product.

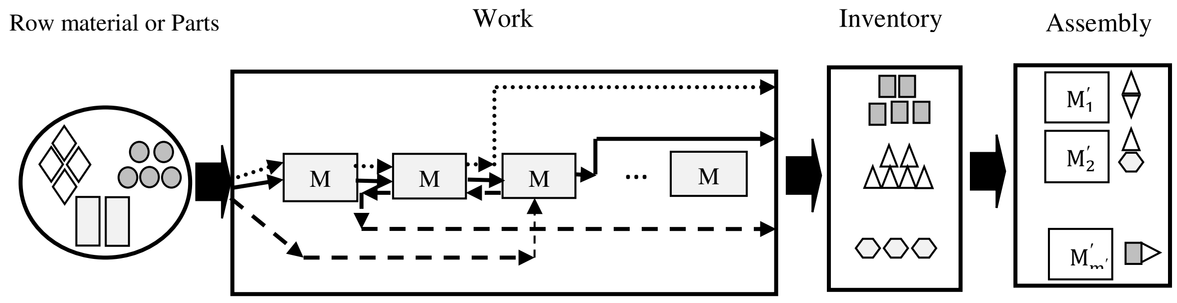

Fig. 1. The layout of the considered problem

2.1. Mathematical model

To better understand the considered problem, its mathematical model is developed in this section. For this model, the following notations and decision variables are used to formulate the problem:

Parameters

P	Total number of products
p	Products index ($p = 1, 2, \ldots, P$)
n	Total number of parts
j	Part index ($j = 1, 2, \ldots, J$)
n_p	Number of sub-jobs of product p
m	Total number of machines at stage 1
i	Machine index at stage 1 ($i = 1, 2, \ldots, m$)
m'	Total number of machines at stage 2
i'	Machine index at stage 2 ($i' = 1, 2, \ldots, m'$)
OHV_{tli}	The l^{th} over hall time duration on machine i
s	Sub-lots index ($j = 1, 2, \ldots, S_{j,p}$)
$O_{s,j,p,h}$	Operation h of sub-lot s of product p's job j
$Pt_{s,j,p,h}$	Time of operation h of sub-lot s of product p's job j
k_i	Number of operations dedicated to machine i
$ps_{j,p,h}$	Processing time of operation h of job j of product p
A_p	Assembly time of product p
$S_{j,p}$	Number of sub-lots of job j of product p
S'_p	Number of sub-lots of product p
$Q_{j,p}$	Lot size of job j of product p
DM_p	Demand for product p
$R_{j,p}$	Ratio of job j to product p
L	A large number
$a_{i,j,p,h}$	1 if $O_{s,j,p,h}$can be performed on machine i; 0 otherwise

Decision variables

C_{max}	Makespan
$F_{s,j,p}$	Completion time of sub-lots of job j of product p
$F'_{s,p}$	Maximum completion time of sub-lots of product p
C_p	Completion time of product p
$t_{s,j,p,h}$	Start time of the processing of operation $O_{s,j,p,h}$
$St_{s',p}$	Starting assembly time of sub-lot s' of product p
$F_{s,j,p,h}$	Completion time of operation $O_{s,j,p,h}$
$Tm_{i,k}$	Start of working time for machine i in priority k
$Sm_{i',k'}$	Start of working time for machine i' in priority k'
SOH_{li}	Start time of the l^{th} over hall on machine i
FOH_{li}	Completion time of the lth over hall on machine i
$U1_{liO_{s,j,p,h}}$	1 if the l^{th} over hall on machine i starts after starting operation $O_{s,j,p,h}$; 0, otherwise
$U2_{liO_{s,j,p,h}}$	1 if the l^{th} over hall on machine i starts before finishing operation $O_{s,j,p,h}$; 0, otherwise
$V1_{liO_{s,j,p,h}}$	1 if the l^{th} over hall on machine i starts before starting operation $O_{s,j,p,h}$; 0, otherwise
$V2_{liO_{s,j,p,h}}$	1 if the l^{th} over hall on machine i finishes after starting operation $O_{s,j,p,h}$; 0, otherwise
$U_{liO_{s,j,p,h}}$	1 if the l^{th} over hall on machine i has interference with operation $O_{s,j,p,h}$; 0, otherwise
$x_{i,s,j,p,h,k}$	1 if operation $O_{s,j,p,h}$ is preformed on machine i in priority k ; 0,otherwise
$Z_{i',s',p,k'}$	1 if sub-lot s' of product p is assembled on machine i' in priority k' ; 0, otherwise
$Q'_{s,j,p,h}$	Size of sub-lot s of job j of product p
$V_{s',p}$	Size of sub-lot s' of product p
$\delta_{s,j,p,h}$	1 if $Q'_{s,j,p,h}$ is more than zero(positive) ; 0, otherwise
$\gamma_{s',p}$	1 if $V_{s',p}$ is more than zero(positive) ; 0, otherwise

Model formulation

$$Min\, Z = (C_{max}) \tag{1}$$

Subject to:

$$c_{max} \geq C_p \qquad \forall p \tag{2}$$

$$t_{s,j,p,h} + ps_{j,p,h}.Q'_{s,j,p,h} \leq t_{s,j,p,h+1} \qquad \forall p,j,s; h = 1,2,\dots,H-1 \tag{3}$$

$$t_{s,j,p,h} + L \times U1_{\mathrm{liO}_{s,j,p,h}} \geq \mathrm{SOH}_{\mathrm{li}} \qquad \forall p,j,s; h = 1,2,\dots,H-1 \tag{4}$$

$$\mathrm{SOH}_{\mathrm{li}} + L \times V1_{\mathrm{liO}_{s,j,p,h}} \geq t_{s,j,p,h} \qquad \forall p,j,s; h = 1,2,\dots,H-1 \tag{5}$$

$$t_{s,j,p,h} + L \times V2_{\mathrm{liO}_{s,j,p,h}} \geq \mathrm{FOH}_{\mathrm{li}} \qquad \forall p,j,s; h = 1,2,\dots,H-1 \tag{6}$$

$$V1_{\mathrm{liO}_{s,j,p,h}} + V2_{\mathrm{liO}_{s,j,p,h}} \leq 1 \qquad \forall p,j,s; h = 1,2,\dots,H-1 \tag{7}$$

$$\mathrm{SOH}_{\mathrm{li}} + L \times U2_{\mathrm{liO}_{s,j,p,h}} \geq t_{s,j,p,h} + ps_{\mathrm{j,p,h}} \qquad \forall p,j,s; h = 1,2,\dots,H-1 \tag{8}$$

$$U_{\mathrm{liO}_{s,j,p,h}} + 1 \geq U1_{\mathrm{liO}_{s,j,p,h}} + U2_{\mathrm{liO}_{s,j,p,h}} \qquad \forall p,j,h,i,s,l \tag{9}$$

$$U_{\mathrm{liO}_{s,j,p,h}} + 1 \geq 2 - U1_{\mathrm{liO}_{s,j,p,h}} + V2_{\mathrm{liO}_{s,j,p,h}} \qquad \forall p,j,h,i,s,l \tag{10}$$

$$F_{s,j,p,h} + L \times \left(1 - U_{\mathrm{liO}_{s,j,p,h}}\right) \geq t_{s,j,p,h} + Pt_{s,j,p,h} + (\mathrm{SOH}_{\mathrm{li}} - \mathrm{FOH}_{\mathrm{li}}) \qquad \forall p,j,h,i,s,l \tag{11}$$

$$t_{s,j,p,h} + ps_{j,p,h}.Q'_{s,j,p,h} \leq t_{s+1,j,p,h} \qquad \forall p,j,h; s = 1,2,\dots,s_{j,p}-1 \tag{12}$$

$$Tm_{i,k} + ps_{j,p,h}.Q'_{s,j,p,h}.x_{i,s,j,p,h,k} \leq Tm_{i,k+1} \qquad \forall p,j,s,h,i;\ \mathrm{k} = 1,2,\dots,k_i-1 \tag{13}$$

$$Tm_{i,k} \leq t_{s,j,p,h} + \left(1 - x_{i,s,j,p,h,k}\right).L \qquad \forall p,j,s,h,k,i \tag{14}$$

$$Tm_{i,k} + \left(1 - x_{i,s,j,p,h,k}\right).L \geq t_{s,j,p,h} \qquad \forall p,j,s,h,k,i \tag{15}$$

$$\sum_p \sum_j \sum_s \sum_h x_{i,s,j,p,h,k} \leq 1 \qquad \forall k,i \tag{16}$$

$$\sum_k x_{i,s,j,p,h,k} = a_{i,j,p,h}\,.\,\delta_{s,j,p,h} \qquad \forall p,j,h,i,s \tag{17}$$

$$t_{s,j,p,h} + ps_{j,p,h}.Q'_{s,j,p,h} \leq F_{s,j,p} \qquad \forall p,j,s,h \tag{18}$$

$$F_{s,j,p} \leq F'_{s,p} \qquad \forall j,p,s \tag{19}$$

$$F'_{s,p} \leq \mathrm{St}_{s',p} \qquad \forall p,s,s' \tag{20}$$

$$\mathrm{St}_{s',p} \leq \mathrm{St}_{s'+1,p} \qquad \forall p,s' \tag{21}$$

$$A_p.V_{s',p} + \mathrm{St}_{s',p} \leq C_p \qquad \forall p \tag{12}$$

$$Sm_{i',k'} + A_p.\mathrm{Z}_{i',s',p,k'}.V_{s',p} \leq Sm_{i',k'+1} \qquad \forall p,i',s';\ k'=1, 2, 3, \dots, k'_{i'}-1 \tag{23}$$

$$Sm_{i',k'} \leq \mathrm{St}_{s',p} + \left(1 - \mathrm{Z}_{i',s',p,k'}\right).L \qquad \forall p,k',i',s' \tag{24}$$

$$Sm_{i',k'} + \left(1 - \mathrm{Z}_{i',s',p,k'}\right).L \geq \mathrm{St}_{s',p} \qquad \forall p,k',i',s' \tag{25}$$

$$\sum_{i'} \sum_{k'} \mathrm{Z}_{i',s',p,k'} = \gamma_{s',p} \qquad \forall p,s' \tag{26}$$

$$\sum_{s=1}^{s_{j,p}} Q'_{s,j,p,h} = Q_{j,p} \qquad \forall p,j \tag{27}$$

$$Q'_{s,j,p,h} \leq Q_{j,p}.\delta_{s,j,p,h} \qquad \forall p,j,s,h \tag{28}$$

$$\delta_{s,j,p,h} \leq Q'_{s,j,p,h} \qquad \forall p,j,s,h \tag{29}$$

$$Q_{j,p} = DM_p.R_{j,p} \qquad \forall p,j \tag{30}$$

$$\delta_{1,j,p,h} = 1 \qquad \forall p,j,h \tag{31}$$

$$\delta_{s+1,j,p,h} \leq \delta_{s,j,p,h} \qquad \forall p,j,s,h \tag{32}$$

$$\sum_{s'} V_{s',p} = DM_p \qquad \forall p \tag{33}$$

$$V_{s',p} \leq \gamma_{s',p}.DM_p \qquad \forall p,s' \tag{34}$$

$$\gamma_{s',p} \leq V_{s',p} \qquad \forall p,s' \tag{35}$$

$$Q'_{1,j,p,H} \geq V_{1,p}.R_{j,p} \qquad \forall p,j,s,s' \tag{36}$$

$$\sum_{ss=1} Q'_{ss,j,p,H} - \sum_{ii=2} V_{ii-1,p}\,.R_{j,p} \geq V_{s',p}.R_{j,p} \qquad \forall p,j,s' = 2,\dots,S'_p, s' = s \tag{37}$$

$$\sum_{ss=2} Q'_{ss-1,j,p,H} - \sum_{ii=2} V_{ii-1,p}\,.R_{j,p} < V_{s',p}.R_{j,p} \qquad \forall p,j,s' = 2,\dots,S'_p, s' = s \tag{38}$$

$$\gamma_{1,p} = 1 \qquad \forall p \tag{39}$$

$$\gamma_{s'+1,p} \leq \gamma_{s',p} \qquad \forall p, s' \qquad (40)$$
$$x_{i,s,j,p,h,k}\ , \mathrm{Z}_{\mathrm{i}',s',\mathrm{p},\mathrm{k}'}\ , \delta_{s,j,p,h}, \gamma_{s',p} \in \{0,1\} \qquad (41)$$
$$C_p \geq 0 \qquad \forall p \qquad (42)$$

The objective function (1) indicates minimization of completion time for all products. Constraint (2) expresses that the makespan is not shorter than the completion time of any product. Constraint (3) enforces each job to follow a predefined processing sequence.
Constraint (4) ensures that if start time of operation $\mathrm{O}_{\mathrm{s,j,p,h}}$ is planed earlier than start time of l^{th} over the hall operation, then the amount of $U1_{\mathrm{liO}_{\mathrm{s,j,p,h}}}$ should be 1. Similarly, Constraint (5) ensures that if start time of operation $\mathrm{O}_{\mathrm{s,j,p,h}}$ is planned after starting of l^{th} over hall operation, then the amount of $V1_{\mathrm{liO}_{\mathrm{s,j,p,h}}}$ should be 1. Constraint (6) shows that if start time of operation $\mathrm{O}_{\mathrm{s,j,p,h}}$ is planned before the end of lth over hall operation, then the amount of $V1_{\mathrm{liO}_{\mathrm{s,j,p,h}}}$ will be 1. Constraint (7) determines that no operation will be done during every over hall. Constraint (8) defines that if the lth over hall operation starts before the end of operation $\mathrm{O}_{\mathrm{s,j,p,h}}$, the amount of $U2_{\mathrm{liO}_{\mathrm{s,j,p,h}}}$ will be 1. Constraint (9) ensures that two parameters $U1_{\mathrm{liO}_{\mathrm{s,j,p,h}}}$ and $U2_{\mathrm{liO}_{\mathrm{s,j,p,h}}}$could not be 1 simultaneously. Constraint (10) determines that if start time of operation $\mathrm{O}_{\mathrm{s,j,p,h}}$ is planned concurrent with the lth over hall operation, $U_{\mathrm{liO}_{\mathrm{s,j,p,h}}}$ should be equal to 1. Equation (11) calculates the finish time of operation $\mathrm{O}_{\mathrm{s,j,p,h}}$ in a condition that this operation interferes in the lth over hall operation. Constraint (12) ensures precedence relationship between sub-lots. Constraint (13) expresses that a machine, at stage one, is not able to process an operation in priority k+1 before processing the operation in priority k. Constraints (14) and (15) force each operation $O_{s,j,p,h}$ which can start after its assigned machine is idle and, thus, previous operation $O_{s,j,p,h-1}$ is completed. Constraints (16) and (17) define the sequence restrictions. Constraint (18) shows the maximum processing time of each sub-lot of each of job for each product. Constraints (19) to (21) define the earliest start time of assembly stage. Constraint (22) shows the completion time of products. Constraint (23) expresses that a machine in the assembly stage, first, assembles the parts of product in priority k'; then, it starts assembling the parts of product in priority k'+1. Constraints (24) and (25) imply that assembling each sub-lot of a product in the second stage can start when its assigned machine is idle and previous product is completed. Constraint (26) assigns each sub-lot of a product to a priority at the second stage. Constraint (27) shows that the total amounts dedicated to the sub-lots of a product's job are equal to those of job's lot size. Constraint (28) ensures that while the amount of $Q'_{\mathrm{s,j,p,h}}$ is not zero, the binary variable $\delta_{\mathrm{s,j,p,h}}$ will be equal to one. However, if $Q'_{\mathrm{s,j,p,h}}$ becomes zero, binary variable $\delta_{\mathrm{s,j,p,h}}$ will be equal to zero according to constraint (29). Constraint (30) defines the amount of each product's jobs (parts) according to the order and the number of jobs needed in each product. Constraints (31) and (32) define the presence of sub-lots and their sizes at the first stage. Constraint (33) shows that the total amounts dedicated to the sub-lots of a product are equal to those of product's demand. Constraints (34-40) also define the presence of sub-lots and their sizes in the assembly stage. Constraint (41) enforces the binary requirements of the decision variables. Constraint (42) implies that the completion time of products must be positive. According to constraints (13) and (23), this model is nonlinear. In these two constraints, a binary variable is multiplied in a continuous variable. So, a linearization method is used as below.

$$Tm_{i,k} + V_{i,s,j,p,h,k} \leq Tm_{i,k+1} \qquad \forall p, j, s, h, i;\ \mathrm{k} = 1,2, \ldots, k_i - 1 \qquad (43)$$
$$V_{i,s,j,p,h,k} \leq ps_{j,p,h}. Q'_{s,j,p,h} \qquad \forall p, j, s, h, i, \mathrm{k} \qquad (44)$$
$$V_{i,s,j,p,h,k} \leq L. x_{i,s,j,p,h,k} \qquad \forall p, j, s, h, i, \mathrm{k} \qquad (45)$$
$$V_{i,s,j,p,h,k} \geq ps_{j,p,h}. Q'_{s,j,p,h} - L. (1 - x_{i,s,j,p,h,k}) \qquad \forall p, j, s, h, i, \mathrm{k} \qquad (46)$$
$$V_{i,s,j,p,h,k} \in \{0,1\} \qquad \forall p, j, s, h, i, \mathrm{k} \qquad (47)$$

2.2. A numerical example

For more understanding of the problem, consider a simple numerical example as shown in Table 2. Assume that we need two operations in job shop and one machine in an assembly stage. Four products of the same kind should be produced and the number of parts is 3. The data for processing time of machining operation, assembly and setup time are given in Table 2. To simplify, two operations are supposed for each job and there is no over hall.
Four solutions are shown in Figures 2, 3, 4, and 5 for this problem. As shown in Figure 2, the operations are partitioned into four blocks and each block consists of the machining operations, the setup operations, and the assembly operations for one product. The completion time of all products is 67 in this plan.

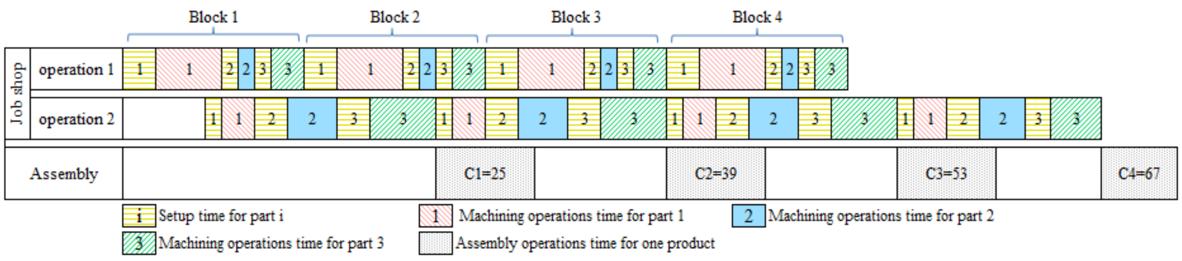

Fig. 2. A schedule for numerical example with C max = 67

In Figure 3, the operations are partitioned into two blocks. Each block consists of the machining operations, the setup operations, and the assembly operations for two products. The completion time of all products is 61 in this plan.

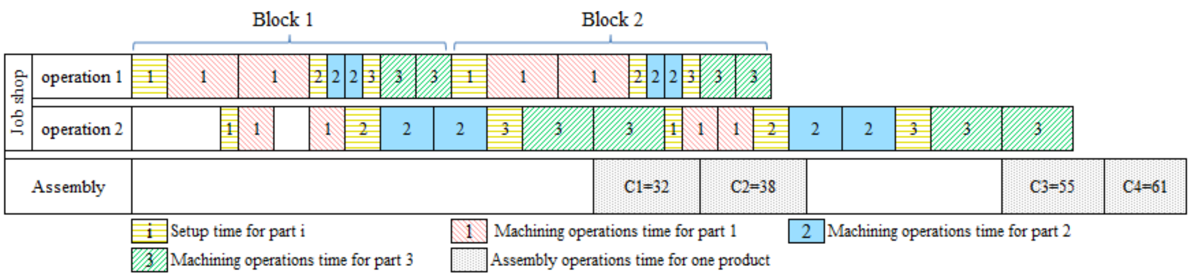

Fig. 3. A schedule of the numerical example with C max = 61

Figure 4 shows another schedule of this problem. According this schedule, the operations are partitioned again into two blocks. However, in this solution, the first block consists of three products and the second block consists of one product. The completion time of all products is also 61 in this plan.

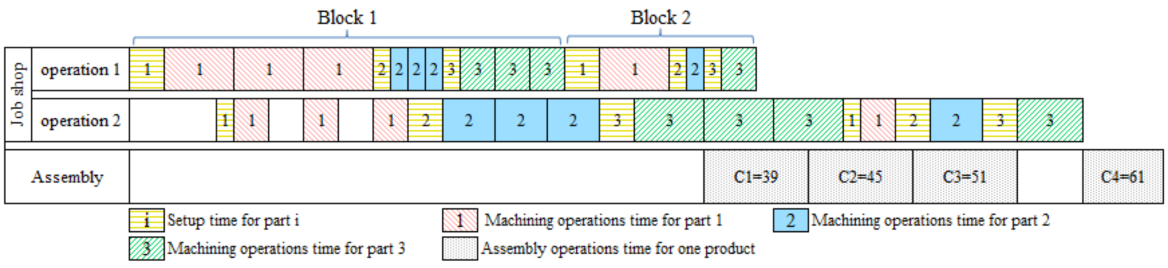

Fig .4. A schedule for numerical example with C max = 61

Figure 5 shows the best schedule in which the blocks are the same as in figure 4; however, the scheduling of parts in blocks is different. The completion time of all products is 52 in this plan.

The yellow color in Table 2 and Fig. 2-5 denotes setup operations, the gray cells denote assembly operations, and the other colors denote process operations of the parts.

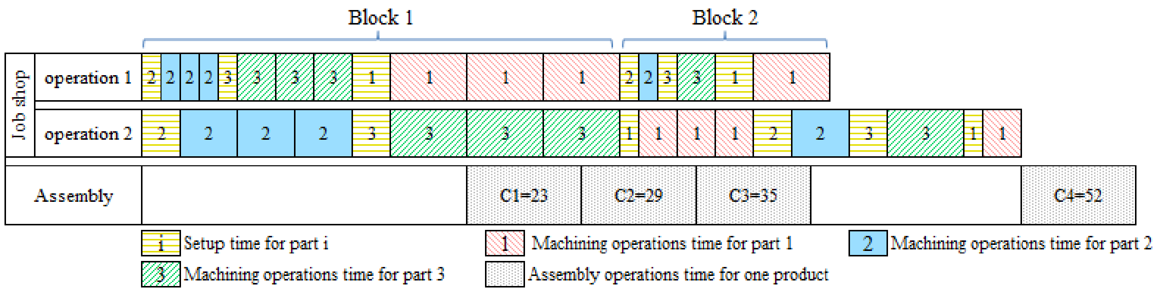

Fig. 5. The best schedule for numerical example with C max = 52

A summary of the results of the sample problem is presented in Table 3. These results indicate that we can improve the objective function of this problem with changing three factors: 1. the number of blocks, 2.the size of blocks, and 3.the sequence of parts in each block.

3. Solution Methods

As mentioned before, the considered problem has been proved to be strongly NP-hard and it requires much time to solve by exact algorithms. Therefore, utilization of heuristics or meta-heuristics approach would be helpful and necessary for medium-sized and large-sized problems. The main decision variables of this problem are the sequence of products to be processed and assembled, the number of lots, the size of each lot and sequence of the parts that will be fixed in all blocks. The assignment parts to machines are ignored because it is assumed that the first idle machine will process the first part that arrives. The notation $N/S/Q$ of Kumar et al. (2000) is modified for the considered problem. This notation is used throughout this paper to indicate the decision methodology used in each of three variables where:

- N is the method used for deciding the number of blocks
- S is the method used for sizing each block
- Q is the method used for sequencing the parts in each block

This notation $N/S/Q$ is modified for the considered problem as $P/N/S/Q$. The new parameter P is defined as the method for sequencing the product.

For example, $Rand/GA/SA/$NEH indicates that sequencing of the product is done randomly, Genetic Algorithm is applied to determine the number of blocks for each product, SA is used to obtain the size of each blocks, and lastly, the sequence of the parts is determined by the NEH algorithm. Therefore, we have a four-step solution approach to the considered problem. These four steps are shown as the solution representation scheme in Figure 6.

The methods that have been developed to solve the problem in four steps based on Figure 6 are described below.

3.1. Extension the johnson algorithm to determine the sequence of products

Fattahi et al. (2014) introduced an extension version of Johnson algorithm for sequencing the product in a hybrid flow shop scheduling followed by an assembly stage. This method is modified for the considered problem that is similar to their study. Therefore, we assume the job shop as the first stage (or first machine) and assembly as the second stage (or second machine). Then, the extended Johnson algorithm is used to determine the sequence of products.

The process time of the first stage consists of the process operations of parts in job shop and is calculated as (48) for any product h.

Algorithm A: $$PT_p = \sum_{h\in n_p} \sum_{j\in n_p} ps_{\mathrm{j,h,p}} \qquad \text{For } p=1,2,\dots,P \qquad (48)$$

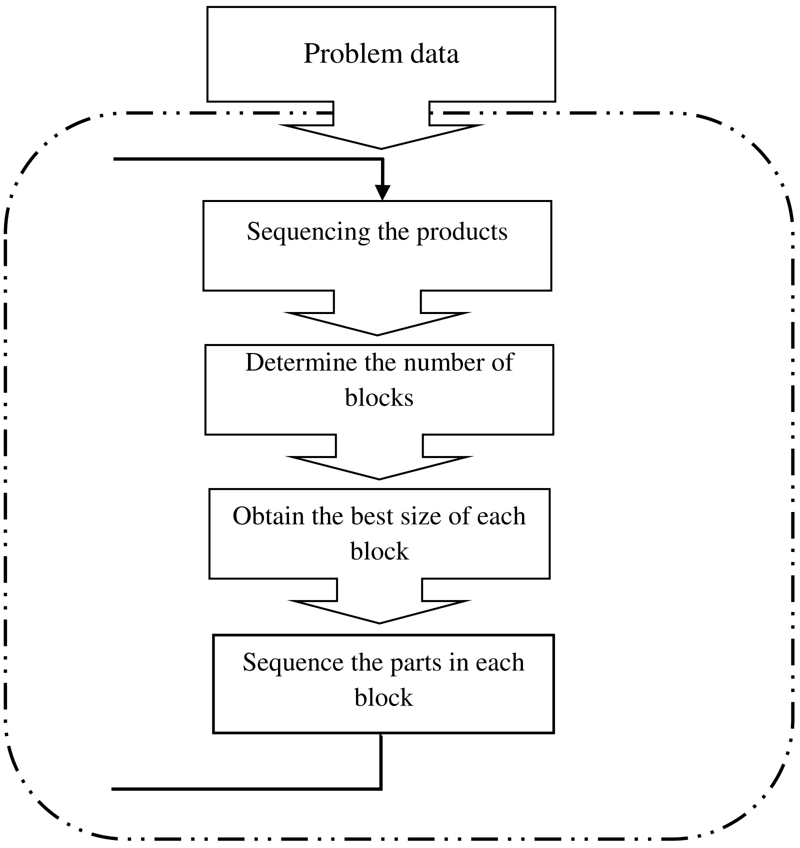
Problem data
Sequencing the products
Determine the number of blocks
Obtain the best size of each block
Sequence the parts in each block

Fig. 6. An overview of the proposed solution methodology

In addition, the assembly time of each product (A_p) is considered as process time in the second stage.
After computing PT_pbased on (48), the sequencing of the products will be done using Johnson algorithm as follows:
1. Determine th PT_p according to equations (48) and A_p from the problem data for all products.
2. Suppose $U = \{p \in P | PT_p < A_p\}$ and $V = \{p \in P | PT_p \geq A_p\}$.
3. Sort the set of U non-decreasing in PT_p and set of V non-increasing in A_p.
4. Determine the sequence of products according to the set of U and V.

3.2. NSH heuristic algorithm for determining the number of blocks

Fattahi et al. (2013) introduced a new heuristic, named SH heuristic algorithm, in order to determine the number of blocks for a hybrid flow shop scheduling problem with setup and assembly operations. This method is developed in this paper for the job shop stage as a new SH heuristic algorithm that is called NSH heuristic algorithm. This algorithm scan the scope of the possible number of blocks with search and test different possible number of blocks. This algorithm is described systematically as below:

```
Step1: consider $B_1 = 1$ (the minimum possible number of blocks) and $B_2$ =number of product $p$ (the maximum possible number of blocks for product $p$) are two amounts for number of blocks and calculate the makespan according them that is called $M_1$and $M_2$ respectively.
Step 2: While (round $|B_1 - B_2|$) > 1, do the following.
If $M_2 \geq M_1$
        $B_2 = (1-\propto) \times B_1 + \propto \times B_2$
                Calculate the new amounts of $M_2$
Else
        $B_1 = \propto \times B_1 + (1-\propto) \times B_2$
                Calculate the new amounts of $M_1$
        End
Step 3: Return the best number of each block $(Min(B_1, B_2))$.
```

Parameter $\propto$ indicates the rate of increasing or decreasing B_1 and B_2 during the algorithm. Based on the large-sized parameter, algorithm does more search; however, the small-sized parameter causes the algorithm to converge and terminate sooner.
Once *NSH* algorithm is run and a new number for blocks are found, two algorithms, based on SA and GA, are used as illustrated in section 3.3 to determine the size of each block. Finally, an algorithm based on *NEH* is used as stated in section 3.4 to sequence the parts of all blocks.

3.3. The proposed algorithms for sizing each block

In order to determine the size of each block, two algorithms based on SA and GA are proposed.

➢ **The proposed algorithm based on SA**

Simulated annealing (SA) is a neighborhood search technique that has produced good results for combinatorial problems. A standard SA procedure begins by generating an initial solution at random. In this section, a simulated annealing algorithm is presented to find good solutions to the block-sizing problem. Therefore, we use a matrix $S_{B\times J}$ to represent the size of blocks. The rows of this matrix show the blocks and its columns show the number of each part j in each block. The members of this matrix should provide the two following conditions:

$$\sum_{b=1}^{B} S_{b,j} = n_p \quad for\, j = 1,2,\dots,J$$

$$\sum_{j=1}^{J} S_{b,j} \geq 1 \quad \text{for } b = 1,2,3,\dots,B$$

The initial block sizing is done with a random operator that assigns a size to each block considering the two above conditions.
The detailed procedure of this algorithm is shown as follows.

```
Step 1: Initialization
Step 1.1: Obtain an initial block sizing ($F^C$ and $F^*$).
Step 1.2: Initiate the initial temperature ($T_0$, $T_C = T_0$), final temperature ($T_f$), cooling rate ($r$), and iterate number in each temperature ($L$).
Step 2: While not yet frozen ($T_C \succ T_f$), do the following.
Step 2.1: Perform the following loop $L$ times.
Step 2.1.1: Done neighborhood search algorithm and select a neighbor $F^{C'}$of $F^C$.
Step 2.1.2: Compute $\triangle = R_{F^{C'}} - R_{F^C}$ (done the computation of makespan)
Step 2.1.3: If $\triangle \leq 0$ then $F^C = F^{C'}$.
Compute $\triangle_b = R_{F^C} - R_{F^*}$.
If $\triangle_b < 0$ set $F^* = F^C$.
Step 2.1.4: If $\triangle > 0$ select a random variable $P \backsim U(0,1)$.
If $e^{-\triangle/T} > P$ set $F^C = F^{C'}$
Step 2.2: Set $T_C = r \times T_C$
Step 3: Return the best solution found for F*.
```

Neighborhood search algorithm

A neighborhood search is proposed in this section. This algorithm has two steps as follows:
Step 1: select two blocks from existing blocks B randomly β_1 , β_2.
Step 2: change the size of blocks β_1 and β_2 and define the new size as follows:

1.	Select $\psi_{\alpha'}$	$1 \leq \psi_{\alpha'} \leq \psi_\alpha + \psi_\beta + \psi_\gamma - 2$
2.	Select $\psi_{\beta'}$	$1 \leq \psi_{\beta'} \leq \psi_\alpha + \psi_\beta + \psi_\gamma - \psi_{\alpha'} - 1$
3.	determine $\psi_{\gamma'}$	$\psi_{\gamma'} = \psi_\alpha + \psi_\beta + \psi_\gamma - \psi_{\alpha'} - \psi_{\beta'}$

Now, there are three new blocks that are different from their old version only in size.

$$\boxed{1}\quad \boxed{2}\quad \cdots \quad \boxed{\alpha'}\quad \cdots \quad \boxed{\beta'}\quad \cdots \quad \boxed{\gamma'}\quad \cdots \quad \boxed{B}$$

In this neighborhood structure, all blocks have a chance to be selected and changed. Also, all neighbors are placed in a feasible region.

The final value of the parameters for this proposed algorithm is obtained using Taguchi settings considering plan of L_n in three levels. In order to determine the best combination of these parameters, three levels of each parameter are examined; finally, the best parameter is obtained as follows:

- ✓ The initial temperature $T_0 = 1000$
- ✓ The final temperature $T_f = 10$
- ✓ The cooling rate $r = 0.95$
- ✓ The number of iteration in each temperature $L = 25$

➢ *The proposed GA algorithm*

A GA (Holland, 1975) is a search technique that imitates the natural selection and biological evolutionary processes. GA has been used in a wide variety of applications, particularly in combinatorial optimization problems which proved to be able to provide near-optimal solutions in a reasonable amount of time (Anandaraman 2011, Luo et al., 2011, and Chakrabortty et al., 2013).

A GA starts with a population of randomly generated candidate solutions (called chromosomes). A chromosome is represented by a string of numbers called genes. Each chromosome in the population is evaluated according to some fitness measures that reflect the objective function value and the satisfaction of problem constraints. The better the fitness values are, the more chances are given to the individual to be selected as a parent. New chromosomes are made of a reproduction operator that usually consists of crossover and mutation procedures. The crossover procedure produces the offspring from two parent individuals by combining and exchanging their elements. Certain pairs of chromosomes are selected on the basis of their fitness. Each of these pairs combines to produce new chromosomes, (offspring) and some of the offspring are randomly modified. The mutation procedure adds small random changes to a chromosome (Maniezzo et al., 2009).

A new population is then formed by replacing some of the original population by an identical number of offspring. The process is repeated until a stopping criterion is met.

In this paper, details of the proposed GA with necessary illustrations are considered as follow.

Each solution (chromosome) is represented as a vector $1 \times \mathrm{B}$ whose elements (genes) are item indices the size of the bth block. An initial population of chromosomes is randomly generated. After some experiments, the best population size is obtained as 20. The evaluation parameter f(c) is the value of objective function. This value is used to measure the fitness of a chromosome. For each partial (chromosome), block sizing is constructed and, accordingly, the makespan of jobs is calculated.

The roulette wheel and tournament selection without replacement is used to select two chromosomes for crossover. Marimuthu et al. (2008) illustrated this selection method in a genetic algorithm for scheduling m-machine flow shop with lot streaming. Accordingly, by using the method used in this paper, at first, the probability of choosing each chromosome $\mathrm{p(c)}$ is estimated as follows:

$$p(c) = \frac{f^*(c)}{\sum_{c=1}^{pop\_size} f^*(c)} \qquad \text{where} \quad f^*(c) = 1/f(c)$$

Then, the cumulative probability $cp(c)$ for all chromosomes is found.

$$cp(c) = \sum_{c=1}^{c} p(c)$$

Then, the random number 'r' between 0 and 1 is spun and a chromosome 'c' is selected, satisfying the following condition:

$$cp(c-1) \le r < cp(c)$$

This selection process is repeated many times equal to parent number needed.

The two-point crossover operator is applied to each pair of parent chromosomes with a probability $\mathrm{p_c}$ (probability of crossover). From sensitivity analysis, probability of crossover is arrived as 0.85.

According to the results of experiments, two kinds of mutation are used in this algorithm. In type 1, the gene being considered is removed from its position and put into another randomly chosen position of the same chromosome with a probability $\mathrm{p_m}/2$ in which $\mathrm{p_m}$ is mutation probability. In type 2, two genes (blocks) α, β are chosen randomly from each chromosome with a probability of $\mathrm{p_m}/2$, and their sizes are changed as below:

$$1 \le \psi_{\alpha'} \le \psi_\alpha + \psi_\beta - 1$$
$$\psi_{\beta'} = \psi_\alpha + \psi_\beta - \psi_{\alpha'}$$

For the considered problem, this mutation scheme works better than other mutation methods such as swapping two genes with another randomly. The probability mutation $\mathrm{p_m}$ is obtained as 0.20 through trials. For each chromosome, a random number is generated ($0 < r < 1$). If the random number is less than 0.1, the chromosome is selected for mutation type 1; if the random number is between 0.1 to 0.2, the chromosome is selected for mutation type 2.

Seventy percent of chromosomes from the present population and thirty percent of chromosomes from children (offspring) are selected as new population and generation. These chromosomes are selected on the basis of their fitness (minimum evaluation makespan). Finally, the stopping criterion is based on the number of iterations without improvement that is considered up to 15 iterations.

3.4. Algorithm NEH for sequencing the parts in each block

Algorithm NEH is used for sequencing the parts in each block, and it is supposed that the sequence of parts is the same in all blocks. According to this algorithm, the parts are sorted in the non-increasing sum of processing time (Nawaz et al., 1983).
The summary of these heuristics and meta-heuristics is shown in Table 4.

4. Test Problems and Computational Results

In order to test the proposed model and evaluate its performance and effectiveness in problem solving, some test problems have been applied in various conditions. The mathematical model is solved by GAMS IDE (ver.23.5) software and the proposed algorithms are coded with MATLAB R2013a software. In order to evaluate the performance of the proposed algorithms, 20 instances are generated randomly. The instances are categorized into three groups (small-, medium-, and large-sized problems) based on the number of operations. For each instance, the algorithms are replicated 10 times. The random sample problems are generated from the distributions in Table 5.
In what follows, some indexes used for analysis are shown.
For comparing the effectiveness of algorithms, Relative Percentage Deviation (RPD) factor is used. RPD factor is formulated as (49) and compares the performance of each procedure to hat of other procedures for each instance:

$$\mathrm{RPD} = \frac{\mathrm{Alg}_{\mathrm{sol}} - \mathrm{Min}_{\mathrm{sol}}}{\mathrm{Min}_{\mathrm{sol}}} * 100 \qquad (49)$$

To evaluate the performance of the proposed algorithm and compare its effectiveness with others, the improvement in initial solutions is measured using a well-known criterion. IMP factor is computed as in Equation (50).

$$IMP = \frac{Alg_{initialsol} - Alg_{finalsol}}{Alg_{finalsol}} * 100 \qquad (50)$$

In addition, a factor named D_{f^*} is used to determine the mean deviation of the best solution. This factor is computed according to Equation (51).

$$D_{f^*} = \frac{\sum_{i=1}^{n}(f_i - f^*)}{n.f^*} \qquad (51)$$

where parameter $f^*$is the best solution obtained by the mathematical model or the algorithms. Therefore, this analysis could only be done for the small-sized problems.
Some other analyses are done on objective function, calculation time, etc.

4.1. Result analysis of small-sized problems

This section presents the results of solving test problems on small scales using a mathematical model and the proposed algorithm. In the small-sized problems, according to the D_{f^*} factor, the proposed algorithms achieved almost good performance. In the comparison of two proposed algorithms, this factor is about on average for algorithm JS/NSH/SA/NEH and for algorithm JS/NSH/GA/NEH.
Figure 7 shows the result of solving the small-sized problems using the mathematical model and two developed algorithms. Based on these results, the algorithm developed based on GA has better performance in solving all small-sized problems.

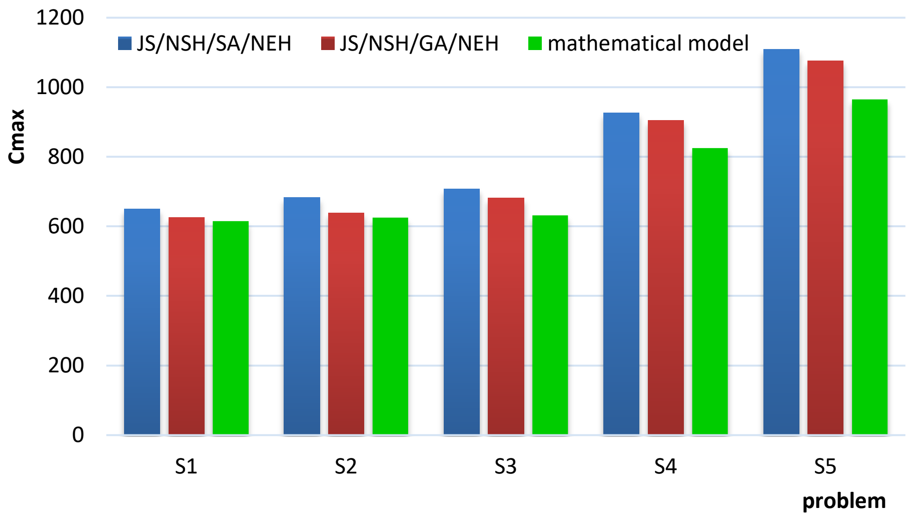

Fig. 7. Performance of proposed algorithm and mathematical model in solving the small-sized problems

In addition, Figure 8 presents the mean deviation of the best solution for two proposed algorithm. As shown in Figure 7, the result of algorithm based on GA has less deviation from the best solution. However, both of two algorithms show good performance in this indicator.

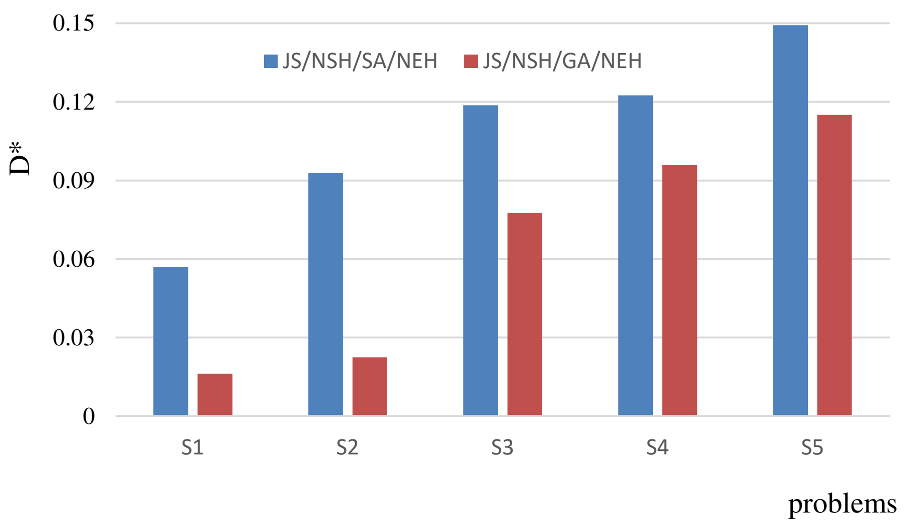

Fig. 8. Comparison of D* in two algorithms

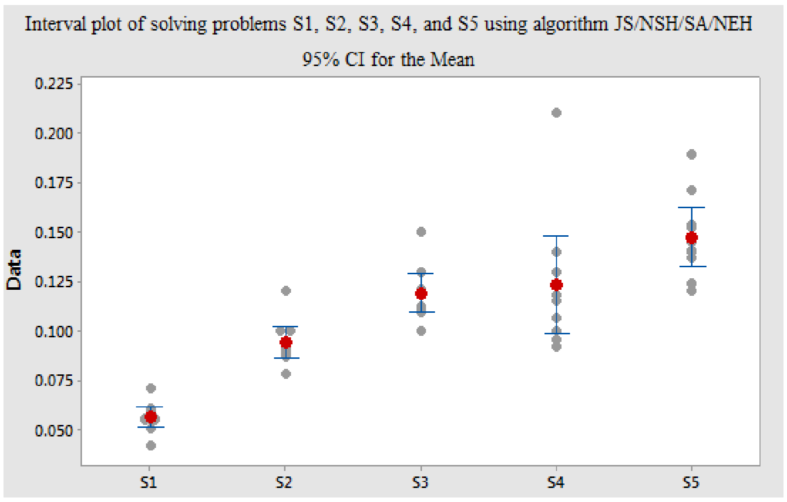

Fig. 9. Interval plot of RPD value for small-sized problems applying algorithm JS/NSH/SA/NEH

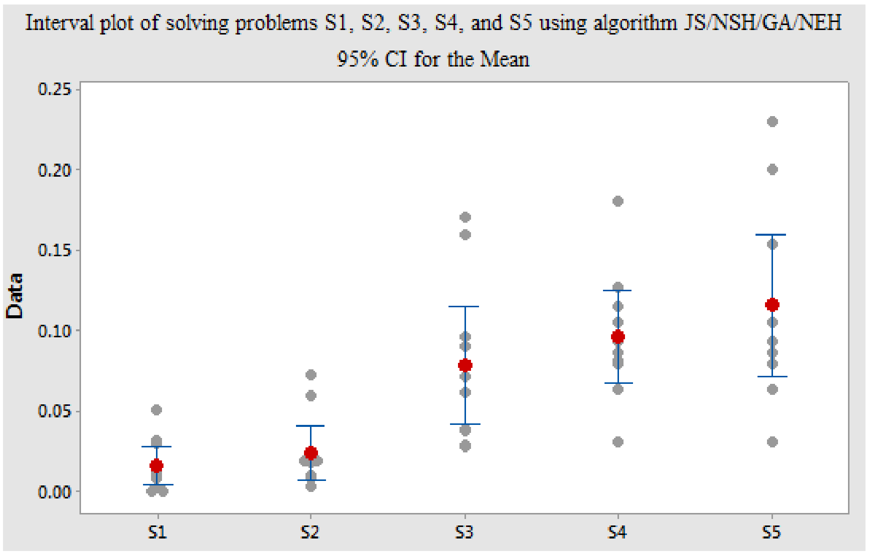

Fig. 10. Interval plot of RPD value for small-sized problems applying algorithm JS/NSH/GA/NEH

The RPD values with 95% confidence interval for the small-sized problems are shown in Figures 9 and 10 for algorithms JS/NSH/SA/NEH and JS/NSH/GA/NEH, respectively. We used the optimum solution obtained from mathematical model as the minimum solution. These figures show the amount of RPD for each run of solving

five problems. Based on this index, the JS/NSH/GA/NEH performs better than the other algorithm again.
These results also represent more convergence for the algorithms JS/NSH/SA/NEH. On the other hand, although algorithm JS/NSH/GA/NEH has better performance in general and on average, but its different runs have more diversity in results.

4.2. *Result analysis on medium and large-sized problems*

In this section, results are presented for all kinds of problems. Figure 11 shows that algorithm JS/NSH/GA/NEH has better performance in all kinds of problems. Preference for this algorithm is clearer, especially in solving the large-sized problems. Performance of the two proposed algorithms in improving the initial solution is also presented in Figure 12. As this figure shows, both of two algorithms could improve the initial solution as appropriate. However, algorithm JS/NSH/GA/NEH is again better in this index. By increasing the size of the problem and the search space, the improvement in initial of JS/NSH/GA/NEH increases. In addition, the lack of improvements in small problems indicated that the algorithms can meet the optimal solution in the first iteration.

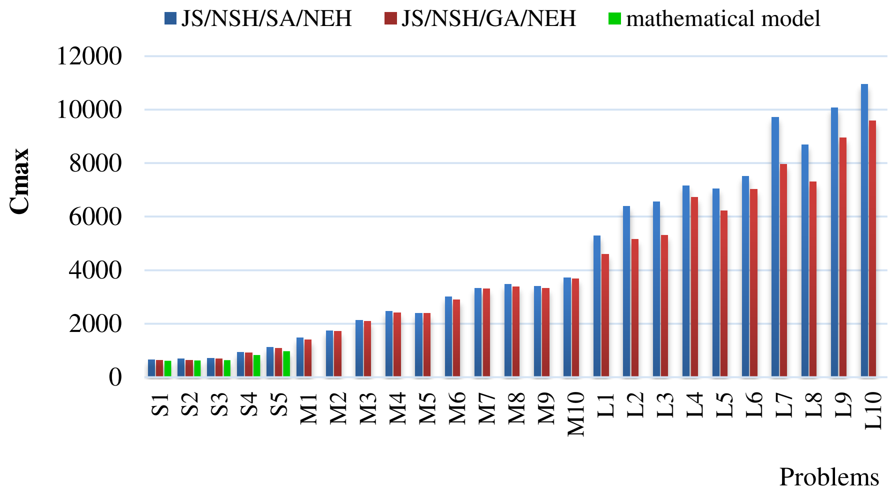

Fig. 11. Performance of proposed algorithms for all of the problems (Makespan)

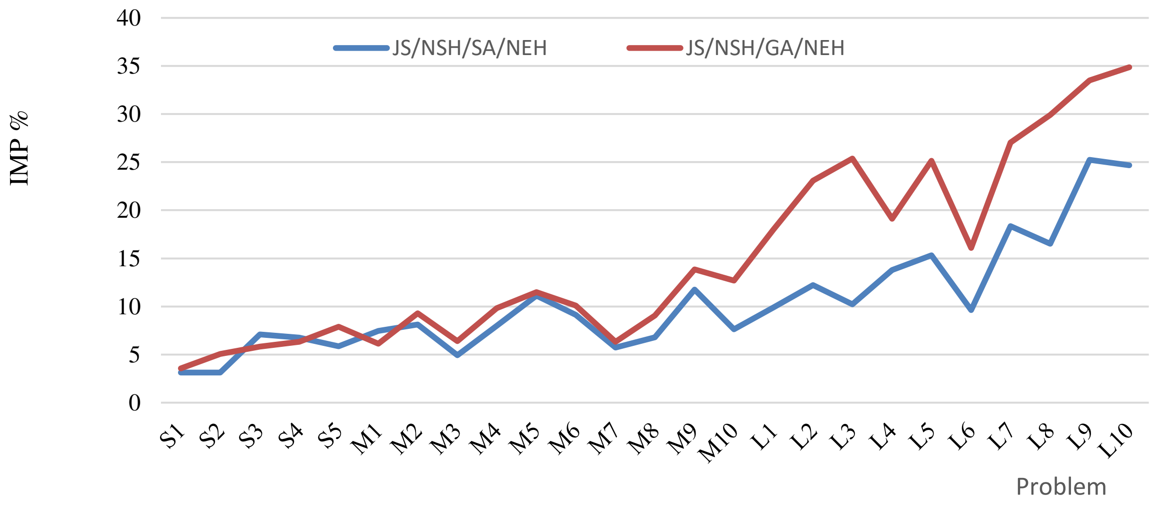

Fig. 12. Performance of proposed algorithms in improvements the initial solution (IMP %)

Trend of improvement is the same for two algorithms; however, the conditions are different for some problems. For example, in solving problem L3, algorithm JS/NSH/GA/NEH could improve the initial solution more than problem L2; however, algorithm JS/NSH/SA/NEH has improved less than problem L2 did. This condition is in reverse with respect to problem L4. These points show situations that one of two algorithm sticks to the local optimum probability.
Finally, due to Figure 13, the algorithm based on SA has presented the final solution sooner than algorithm GA. This difference is clear especially in solving the large-sized problems.

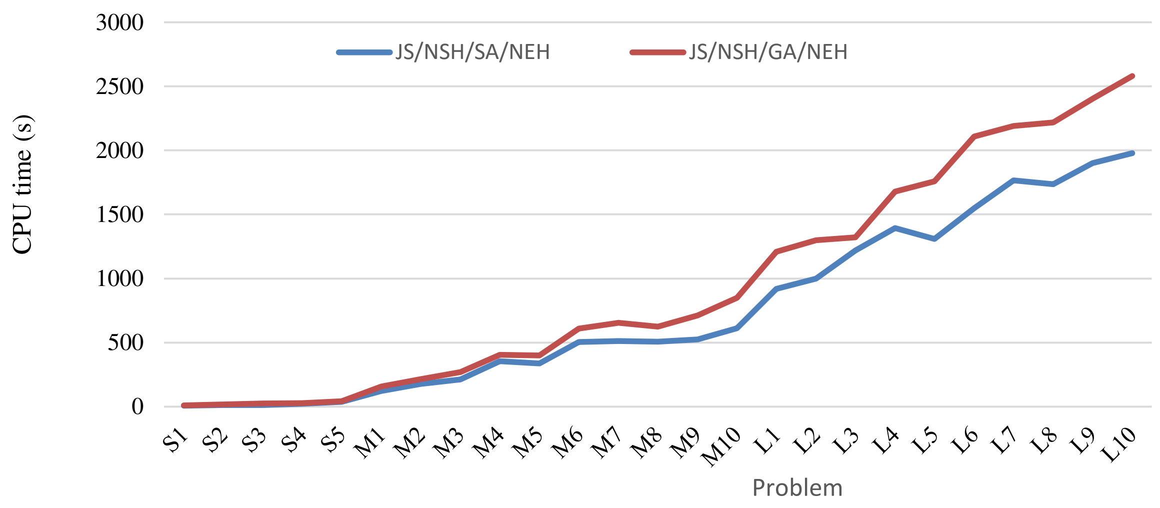

Fig. 13. Computational time of two algorithms CPU time (s)

5. Conclusion

A two-stage production system was studied in this paper. In this system, a number of products of different kinds are ordered to be produced. Each product is assembled with a set of several parts. The first stage is a job shop to produce parts and the second stage is an assembly shop with several parallel machines. Maintenance operations and access restrictions to machines were considered in the first stage. The objective function was to minimize the completion time of all products (makespan). This kind of production system has many applications in industry and, thus, has been studied by many researchers during three last decades.

At first, this problem was described and modelled as a mixed integer linear programming. Since this problem has been proved to be strongly NP-hard, two new algorithms based on heuristic, GA, and SA were developed to solve the medium- and large-sized problems. For evaluation of the effectiveness of the proposed algorithms, a statistical analysis was used along with some factors such as Relative Percentage Deviation (RPD) and well-known criterion IMP. Various problems were designed and solved by the proposed algorithms. Computational results revealed that both two proposed algorithms have good performance. However, the method based on a genetic algorithm performs better than the other proposed algorithm with respect to the objective function. However, the algorithm based on SA obtained the final solution in a shorter amount of time.

In solving the small-sized problems, both of two algorithms presented less than 15% deviation from the global optimum. In addition, both of the two proposed algorithms show a good improvement in initial solution. This improvement increases more than 30% for the large-sized problems.

For future study, this problem can be considered under the condition of uncertainty for the parameters. Another study could be solving the same problem by developing the other meta-heuristic algorithms and comparing results with the proposed algorithms in this paper.

References

Al-Anzi, F.S. &Allahverdi, A. (2013). An artificial immune system heuristic for two-stage multi-machine assembly scheduling problem to minimize total completion time. *Journal of Manufacturing Systems,* 32 (4), 825– 830.

Allahverdi A, Al-Anzi F. S (2009). The two-stage assembly scheduling problem to minimize total completion time with setup times. *Computers & Operations Research,* 36:2740-2747

Berrichi, A., Amodeo, L., Yalaoui, F., Châtelet, E., Mezghiche, M. (2008). Bi-objective optimization algorithms for joint production and maintenance scheduling: application to the parallel machine problem. *Journal of Intelligent Manufacturing*, 20, 389-400.

Buscher, U., & Shen, L. (2009). An integrated tabu search algorithm for the lot streaming problem in job shops. European *Journal of Operational Research*, 199 (2), 385-399.

Cui, W., Lu, Z., Li, C., & Han, X. (2018). A proactive approach to solve integrated production scheduling and maintenance planning problem in flow shops. *Computers & Industrial Engineering* 115 (2018) 342–353.

Cummings, D.H., & Egbelu, P.J. (1998). Minimizing production flow time in a process and assembly job shop. *International Journal of Production Research*, 36(8), 2315–2332.

Chan, F.T.S., Wong, T.C., & Chan, L.Y. (2008). Lot streaming for product assembly in job shop environment. *Robotics and Computer-Integrated Manufacturing,* 24 (3), 321–331.

Chan, F.T.S., Wong, T.C., &Chan, L.Y. (2009). An evolutionary algorithm for assembly job shop with part sharing. *Computers & Industrial Engineering*, 57 (3), 641–651.

Chen, J., & Steiner, G. (1997). Lot streaming with detached setups in three-machine flow shops. *European Journal of Operational Research.* 96(3), 591-611.

Daneshamooz, F., Jabbari, M., &Fattahi, P. (2013). A model for job shop scheduling with a parallel assembly stage to minimize makespan. *Journal of Industrial Engineering Research in Production Systems,* 2(4), 39-53.

Dauzere - Peres, S., & Lasserre, J. B. (1993). An iterative procedure for lot streaming in job-shop scheduling. *Computers & Industrial Engineering*, 25 (4), 231-234.

Demir, Y., &Isleyen, S.K. (2014). An effective genetic algorithm for flexible job-shop scheduling with overlapping in operations. *International Journal of Production Research*, 52(13), 3905-3921.

Eschelman, L., Caruana, R., &Schaffer, D. (1989). Biases in the crossover landscape. Proc. Third international conference on genetic algorithms, Morgan Kaufman Publishing, 21-29.

Fattahi, P., Hosseini, S.M.H., &Jolai, F. (2013). A mathematical model and extension algorithm for assembly flexible flow shop scheduling problem. *International Journal of Advanced Manufacturing Technology*, 65 (5), 787-802.

Fattahi, P., Daneshamooz, F. (2017). Hybrid algorithms for job shop scheduling problem with lot streaming and a parallel assembly stage. *Journal of Industrial and Systems Engineering*, 10: 92-112.

Garey, M.R., Johnson, D.S., & sethi, R. (1976). The Complexity of flow shop and job shop scheduling. Mathematics of Operation Research, 1 (2), 117-129.

Jeong, H., Park, J., &Leachman, R.C. (1999). A batch splitting method for a job shop scheduling problem in an MRP environment. *International Journal of Production Research,* 37 (15), 3583-3598.

Khoukhi, F.E., Boukachour, J., Hilali Alaoui, A.E., (2017). The “Dual-Ants Colony”: A Novel Hybrid Approach for the Flexible Job Shop Scheduling Problem with Preventive Maintenance. Computers & Industrial Engineering, 106, 236-255.

Koulamas Ch, Kyparisis G. J (2001). The three-stage assembly flow shop scheduling problem. *Computers & Operations Research* 28:689-704

Krikpatrick, S., Gelatt, C.D., &Vecchi, M.P. (1983). Optimization by Simulated Annealing. Science, 220 (4598), 671-680.

Lee, C.Y., Cheng, T.C.E., &Lin, B.M.T. (1993). Minimizing the makespan in the 3-machine assembly-type flow shop scheduling problem. *Management Science*, 39 (5), 616-625.

Lei, D., &Guo, X. (2013). Scheduling job shop with lot streaming and transportation through a modified artificial bee colony. *International Journal of Production Research,* 51(16), 4930-4941.

Maleki-Darounkolaei, A., Modiri, M., Tavakkoli-Moghadam, R., &Seyyedi, I. (2012). A three-stage assembly flow shop scheduling problem with blocking and sequence depended setup times. *Journal of Industrial Engineering International*, 8:26.

Manne, A.S. (1960). On the job shop scheduling problem. *Operational Research,* 8 (2), 219-223.

Mohammadi, E. (2016). Multi objective job shop scheduling problem with an assembly stage and lot streaming .Master of Science Thesis, Buali-Sina University.

Mokhtari, H., Dadgar, M. (2015). Scheduling optimization of a stochastic flexible job-shop system with time-varying machine failure rate. *Computers & Operations Research*, 61: 31-45.

Navaei, J., Fatemi-Ghomi, S.M.T., Jolai, F., &Mozdgir, A. (2014). Heuristics for an assembly flowshop with non-identical assembly machines and sequence dependent setup times to minimize sum of holding and delay costs. *Computers & Operations Research,* 44, 52–65.

Nejati, M., Mahdavi, I., Hassanzadeh, R., &Mahdavi-Amiri, N. (2016). Lot streaming in a two-stage assembly hybrid flow shop scheduling problem with a work shift constraint. *International Journal of Production Research*, 33(7), 459-471.

Potts C.N, Sevast'Janov S.V, Strusevich V. A, Van Wassenhove L.N, Zwaneveld C.M (1995). The two-stage assembly scheduling problem: Complexity and approximation. Operations Research 43:346-355.

Reiter, S. (1966). A system for managing job-shop production. *Journal of Business,* 39 (3), 371–393.

Seyedi, I., Maleki-Daronkolaei, A., &Kalashi, F. (2012). Tabu search and simulated annealing for new three-stage assembly flow shop scheduling with blocking. Interdisciplinary *Journal of Contemporary Research In Business,* 4 (8), 394-402.

Spears, W. M., &De Jong, K. A. (1991). On the virtues of uniform crossover. Proceedings of the Fourth International Conference on Genetic Algorithms, 230-236.

Wagner, B.J., &Ragatz, G. (1994). The impact of lot splitting on due date performance. *Journal of Operations Management,* 12 (1), 13-25.

Wagner, H. (1959). An integer linear-programming model for machine scheduling. Naval Research logistics Quarterly, 6 (2), 131-140.

Wang, S., & Liu, M. (2014). Two-stage hybrid flow shop scheduling with preventive maintenance using multi-objective tabu search method. *International Journal of Production Research*, 52(5), 1495–1508.

Wong, T.C., Chan, F.T.S., &Chan, L.Y. (2009). A resource-constrained assembly job shop scheduling problem with Lot Streaming technique. *Computers & Industrial Engineering,* 57 (3), 983–995.

Wong, T.C., &Ngan, S.C. (2013). A comparison of hybrid genetic algorithm and hybrid particle swarm optimization to minimize makespan for assembly job shop. Applied Soft Computing, 13(3), 1391–1399.

Xiao, L., Song, S., Chen, X., and Coit, D.W. (2016). Joint optimization of production scheduling and machine group preventive maintenance. *Reliability Engineering & System Safety*, 146, 68–78.

Xiong, F., Xing, K., &Wang, F. (2015). Scheduling a hybrid assembly-differentiation flow shop to minimize total flow time. European *Journal of Operational Research,* 240, 338-354

Yao L., &Sarin, C.S. (2014). Multiple-Lot Lot Streaming in a Two-stage Assembly System. Essays inProduction Project Planning and Scheduling, Springer US.

Yazdani, M., Amiri, M., &Zandieh, M. (2010). Flexible job shop scheduling with parallel variable neighborhood search algorithm. E*xpert system with applications*, 37(1), 678-687.

Yokoyama, M., &Santos, D.L. (2005). Three-stage flow-shop scheduling with assembly operations to minimize the weighted sum of product completion times. European *Journal of Operational Research*, 161, 754–770.

Zhang, R., &Cheng, W. (2011). A simulated annealing algorithm based on blocking properties for the job shop scheduling problem with total weighted tardiness objective. *Computer and operation research,* 38 (5), 854-867.

This article can be cited: Hosseini S.M.H.(2019). Modelling and Solving the Job Shop Scheduling Problem Followed by an Assembly Stage Considering Maintenance Operations and Access Restrictions to Machines. *Journal of Optimization in Industrial Engineering*. 12 (1), 63- 78.
http://www.qjie.ir/article_539595.html
DOI: 10.22094/JOIE.2018.760.1484