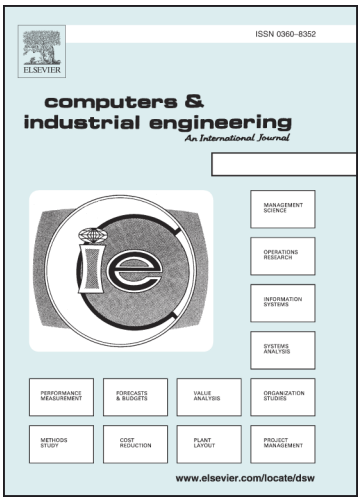
A two-stage ant colony algorithm for hybrid flow shop scheduling with lot sizing and calendar constraints in printed circuit board assembly

Journal Pre-proofs



A two-stage ant colony algorithm for hybrid flow shop scheduling with lot sizing and calendar constraints in printed circuit board assembly

Wei Qin, Zilong Zhuang, Yang Liu, Ou Tang

PII: S0360-8352(19)30584-4
DOI: https://doi.org/10.1016/j.cie.2019.106115
Reference: CAIE 106115

To appear in: *Computers & Industrial Engineering*

Received Date: 27 April 2019
Revised Date: 14 August 2019
Accepted Date: 5 October 2019

Please cite this article as: Qin, W., Zhuang, Z., Liu, Y., Tang, O., A two-stage ant colony algorithm for hybrid flow shop scheduling with lot sizing and calendar constraints in printed circuit board assembly, *Computers & Industrial Engineering* (2019), doi: https://doi.org/10.1016/j.cie.2019.106115

A two-stage ant colony algorithm for hybrid flow shop scheduling with lot sizing and calendar constraints in printed circuit board assembly

Wei Qin [a, *], Zilong Zhuang [a], Yang Liu [b,c, *], Ou Tang [b]

[a] School of Mechanical Engineering, Shanghai Jiao Tong University, Shanghai, China

[b] Department of Management and Engineering, Linköping University, SE-581 83 Linköping, Sweden

[c] Department of Production, University of Vaasa, 65200 Vaasa, Finland

[*] ***Corresponding author***: wqin@sjtu.edu.cn; yang.liu@liu.se

Highlights

- PCB assembly shop scheduling problem has many challenging characteristics.
- A hierarchical approach is developed to decompose the original problem.
- The sub-problems include job sequencing and lot scheduling with lot sizing.
- A two-stage ant colony algorithm with lot sizing is proposed to evolve best results.

A two-stage ant colony algorithm for hybrid flow shop scheduling with lot sizing and calendar constraints in printed circuit board assembly

Abstract: This paper investigates a multi-stage hybrid flow shop scheduling problem with challenging characteristics including lot sizing, calendar constraints and sequence-dependent setup times in a real-world printed circuit board (PCB) assembly shop. Besides, other characteristics such as unrelated parallel machines and stage skipping also complicate the problem. Such features make the scheduling problem very difficult to find an optimal or near optimal solution. To reduce the complexity of such a PCB scheduling problem, this study develops a hierarchical approach which decomposes the original problem into two highly coupled sub-problems including job sequencing and lot scheduling with lot sizing, and further proposes a two-stage ant colony algorithm with lot sizing to evolve best results in the makespan performance. Extensive computational experiments have been conducted to illustrate the superiority of the algorithm in terms of computational time and stability.

Keywords: *hybrid flow shop; PCB assembly shop scheduling; ant colony algorithm; lot sizing*

1 Introduction

Printed circuit boards (PCBs) have been widely used as components for electrical/electronic devices (Noroozi and Mokhtari, 2015). In the past ten years, Mainland China has become the largest manufacturing region of PCBs in the world (Yang, 2018). However, most of the PCB manufacturers are facing the challenge of improving the production efficiency in order to cope with the fierce competition. Therefore, motivated by a real-life problem in a semiconductor enterprise in Suzhou,

this study aims to examine the scheduling decision so as to enhance resource utilization, reduce lead times and subsequently improve production efficiency.

PCB assembly is the process of placing electronic components (resistors, capacitors, transistors, integrated circuits, etc.) of different shapes and sizes at pre-specified locations on a bare board (Castellani et al. 2019). According to the definition of operations scheduling, the PCB assembly process in this factory can be viewed as a hybrid flow shop (HFS) scheduling problem, also considered as a flexible flow line or a flow shop with multiple machines on some/all production stages. According to the classic definition of HHS, the machines of each stage are unrelated and in parallel, which means that a job can be processed on any of those machines and the processing times at each stage are different. The flow of jobs through the HFS is unidirectional. Each job is processed by one machine at each stage and it must go through all stages (Linn and Zhang, 1999).

Different from an ordinary HFS, some characteristics in our case in PCB assembly shops significantly substantiated the complexity of the scheduling problem. The distinctive features are summarized as follows:

(1) **Lot sizing**. Considering the machine setup times which are required between processing of different PCBs, the orders are always split into a small number of batches. PCBs are produced in batches in order to enhance the utilization of machines and assembly lines. Each batch consists of identical PCBs with the same due date. All batches within the same order have the same sequence of stations to visit since the PCBs have the same route. Each batch of one PCB type is processed without stopping, that is, it cannot be preempted by another PCB type until the batch is completed.

(2) **Calendar constraints**. The calendar is a tool to set the work shifts of all machines. The work shift is a segment of continuous available times of a machine. In the investigated PCB assembly shop, this means the machines such as screen printers, refold machines, inspection equipment, and also different types of placement machines are available only during working times in the calendar.

(3) **Setup times/process time/stage skipping.** Besides, the setup times for the jobs depend on sequences of PCBs to be processed, whereas the machines at each stage are unrelated, i.e., the processing times for different machines to complete the same batch are different. The stage skipping is also allowed because not all jobs are required to go through all stages.

To our best knowledge, these factors, such as lot sizing, calendar constraints and sequence dependent setup times are generally considered separately. In a hybrid flow shop environment, lot sizing has been considered to enhance the production efficiency or reduce energy costs (Chen et al. 2018; Meng et al. 2018; Zohali et al. 2019). Similarly, hybrid flow shop scheduling with unavailability constraints has attracted the attention of a lot of researchers (Shoaardebili and Fattahi, 2015; Bozorgirad and Logendran, 2016; Cui et al. 2016). Besides, hybrid flow shop scheduling with sequence dependent setup times is a very important research field and many research papers have been published (Hatami et al. 2015; Xu et al. 2017; Pan et al. 2017). However, few researchers discussed the combination of these factors in one model. The purpose of this paper is to formulate the problem that considers practical conditions and constraints, and to develop a solution procedure that provides good results. The practical use of scheduling technique for this type of problem is still rare (Arora and Agarwal, 2016; Rossit et al. 2018; Lee and Loong, 2019).

This study is an attempt to narrow the gaps between theoretical development and industrial practices in such a complicated PCB assembly scheduling problem by taking into account the above-mentioned operational characteristics jointly in one model. The remainder of this paper is organized as follows. In Section 2, we present a brief literature review. In Section 3, we provide the description and formulation of the investigated scheduling problem. Section 4 describes the basic ant colony algorithm and summarizes the process of formulation and the main characteristics of the proposed algorithm. Section 5 presents the outcomes of the experiment study. Finally, some concluding remarks are given in Section 6.

2 Literature review

The PCB scheduling problem considered in this paper can be described based on the literature along the following areas: (1) PCB assembly shop scheduling; (2) scheduling with lot sizing; and (3) scheduling with limited machine availability.

2.1 PCB assembly shop scheduling

Many factors affect the efficiency of PCB assembly, such as customer orders, component allocation, PCB grouping, component sequence and feeder arrangement. Many researchers have developed different algorithms to optimize different factors in the PCB assembly. Krishnan (2014) studied placement optimization in a PCB assembly process, and proposed a neighbourhood search heuristic to optimize the component placement sequence and minimize the placement time. Alkaya and Duman (2015) adopted a metaheuristic approach to optimize the chip shooter component placement machines by decomposing the problem into placement sequencing problem and feeder configuration problem. Raduly-Baka et al. (2017) proposed two heuristics based on job grouping to optimize the problem with the idea of constructing the minimum number of component reel modules. Navaei and ElMaraghy (2017) developed a new grouping and sequencing policy for finding the optimal sequence of various product variants to improve machine utilisation. Tóth et al. (2018) presented a two-step optimisation method for the machine reconfiguration and workload balancing in the case of multiple PCB batches of different sizes and PCB types.

2.2 Scheduling with lot sizing

The technique of lot sizing (lot streaming) has been widely used in scheduling in recent years. Cheng et al. (2013) summarized that lot streaming has been applied for the processing of jobs in a variety of machine configurations, including flow shops, job shops, open shops, parallel machines, and hybrid flow shops. Chakaravarthy et al. (2014) proposed improved sheep flock heredity algorithm and artificial bee colony algorithms to solve n-jobs, m-machines lot streaming problem in a flow shop with an

equal size of sub-lots. Nejati et al. (2016) addressed the two-stage assembly scheduling problem with m machines at the first stage and n assembly machines at the second stage with the consideration of lot sizing issue, and proposed a genetic algorithm and simulated annealing approach to optimize the two-stage assembly hybrid flow shop problem. Huang and Yu (2017) developed an improved ant colony optimization to resolve multi-objective job shop scheduling problem with equal-size lot splitting. Wang et al. (2019) studied integrated batching and lot streaming problem with variable sub-lots, incompatible job families, and sequence-dependent setup times, and developed heuristics for an efficient solution. From the literature we can see that most studies only indicate the condition of job grouping or assume that job batches already exist, whereas very few studies have addressed how to decide the job batches.

2.3 Scheduling with limited machine availability

The traditional scheduling problem assumes that machines are continuously available for processing throughout the planning horizon. However, in industrial practice, this availability may not be true due to e.g. a machine breakdown (stochastic) or preventive maintenance or calendar capacity (deterministic). Seidgar et al. (2016) employed three multi-objective optimization methods, namely fast non-dominated sorting genetic algorithm, multi-objective imperialist competitive algorithm, and non-dominated ranking genetic algorithm to find the pareto-optimal front for large sized two-stage assembly flow shop scheduling problem with preventive maintenance activities. Seidgar et al. (2017) investigated a two-stage assembly flow shop problem which considers machines breakdown, and presented a genetic algorithm and new self-adapted differential evolutionary algorithm. Cui et al. (2017) proposed a proactive approach to deal with the integration of production scheduling and maintenance planning in flow shops. González-Neira et al. (2017) provided a general overview of the flow shop scheduling problems under uncertainties. Han et al. (2018) proposed an evolutionary multi-objective robust scheduling algorithm for blocking lot-streaming flow shop scheduling problems with machine breakdowns.

Based on the above brief literature review, we find that although a broad literature on PCB assembly shop scheduling has been published, only few studies have dealt with the hybrid flow shop scheduling problem with a joint consideration of lot sizing and calendar constraints, which are real-life restrictions in PCB assembly shops and should not be simplified or ignored. This study therefore is an attempt to bridge this gap by proposing a comprehensive solution to this complicated scheduling problem which coordinates all important characteristics. In this study, we develop a hierarchical approach which decomposes the original problem into two highly coupled sub-problems including job sequencing and lot scheduling with lot sizing, and further propose a two-stage ant colony algorithm with lot sizing to evolve best results in the makespan performance.

3 Problem formulation

3.1 Description of PCB assembly shop scheduling

The layout detail of the assembly shop in the case company is depicted in Fig.1. All the PCBs pass through 5 stages of manufacturing (assembly) process: surface mounting, reflow soldering, automatic/manual insertion, wave soldering and inspection by burn-in testing. The PCBs composed by small electronic elements might skip the insertion stage. After the final test, the PCBs will go into assembly lines for the final products.

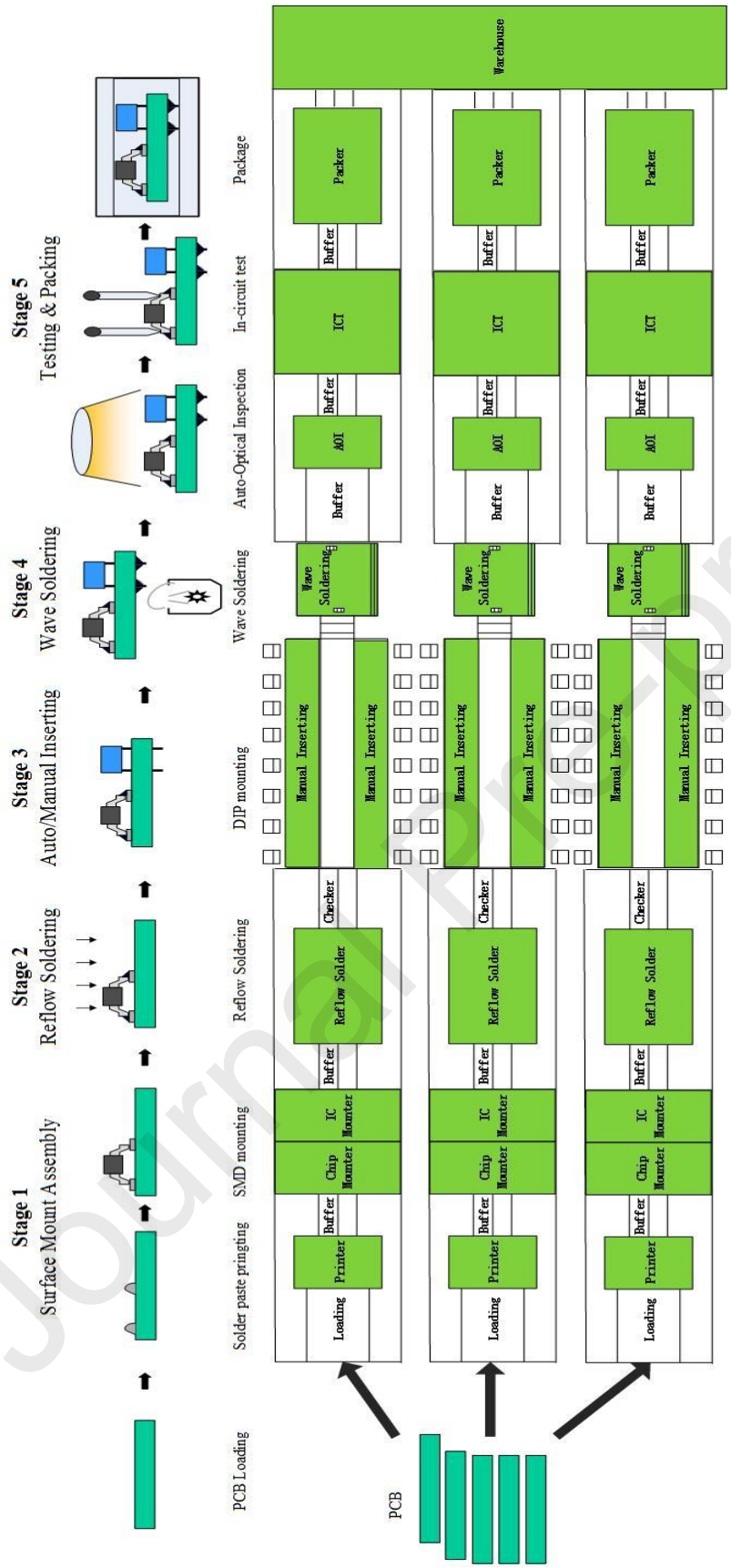


Fig.1 Layout of the PCB assembly shop

Suppose that there are *n* jobs and *m* stages in the PCB assembly shop where each job is available at time zero and contains lot of items of the same kind. Each stage can be processed by several machines, which are capacitated. The capacity, which is limited, will be consumed in producing items. The target is to minimize the makespan. Assumptions and constraints for the scheduling problem are as follows:

1) All jobs are independent and available for processing at the initial time of scheduling.
2) Each job contains identical items with the same ready time and due date and can be separated into several sub-lots.
3) One machine can process only one job at a time and one job can be processed by only one machine at any time.
4) All the items in a sub-lot should be processed on the same machine.
5) A new sub-lot can start only after the completion of the previous sub-lot produced on the same machine, i.e., preemption is not allowed.
6) The machines at every stage are unrelated, i.e., the processing times of different machines to complete the same sub-lot are different.
7) Not all jobs are required to go through all stages, e.g., some jobs might skip some production stages.

Fig.2 (a) shows an example of scheduling of 2 jobs, each with 2 items, being processed through 3 stages. Each stage of the production system contains 2 unrelated parallel machines capacitated by calendar. Job1 can skip stage 2. In this particular case, we assume that a sub-lot contains only one item. The grey fields in the Gantt chart represent the available times for each machine and a sub-lot is marked by two digits composed by job and sub-lot number respectively. For instance, sub-lot "12" means the second sub-lot of job 1.

As reviewed previously, scheduling with calendar constraints was poorly studied. Fig.2 (b) shows the traditional lot scheduling approach. Traditionally when we assign a job with lots of items to a machine, we first split the job to several sub-lots according to some criteria, then we obtain a sequence of the sub-lots and assign the sub-lots to the machine in continuous time axis. In this paper, we adopt an

approach named "lot scheduling with capacity". As the available time for a machine is not continuous in the investigated problem, we consider the capacities, segments of continuous time, as the research objects. Specifically, we sequence the capacities and assign the capacities to the jobs according to capacity sequence until all the items are assigned. As shown in Fig.2 (c), the grey fields are available times (capacities) of machine M1. Suppose we take the capacity sequence "213", then the capacity 2 will be assigned to the job first, followed by capacity 1, and finally capability 3. As depicted in the figure, capacity 2 cannot arrange all the items, therefore, we split these items (which are originally assigned to capacity 2) to form a sub-lot, named sub-lot 1, and assign the rest of the job to capacity 1 and 3. We repeat this process till all items are assigned with capacity.

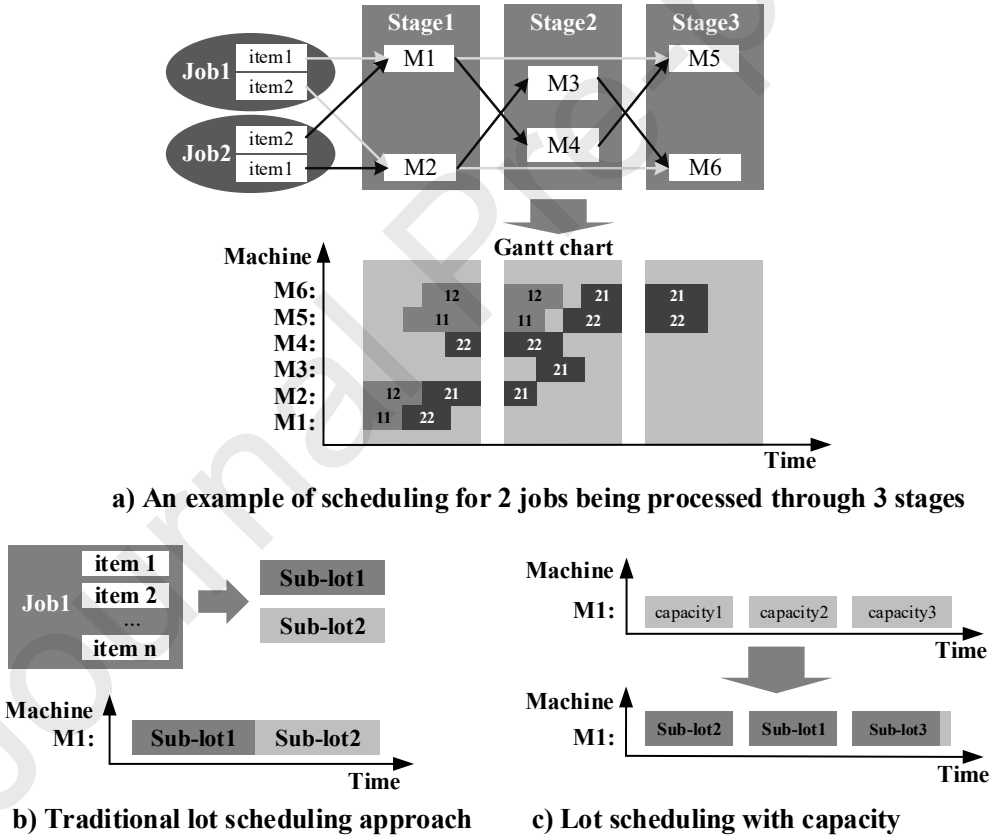


Fig.2 Examples of the PCB assembly shop scheduling

3.2 Mathematical formulation

Now a complete mathematical model is developed with the consideration of all issues mentioned above. Suppose that there are totally n jobs $\{J_1, J_2, \ldots, J_n\}$ to be

scheduled. Each job J_i needs maximum m stages to be completed where stage j of job J_i is denoted $O_{i,j}$ and $O_{i,j}$ can be separated to $L_{i,j}$ sub-lots where sub-lot k of stage $O_{i,j}$ is denoted by $B_{i,j,k}$. Each stage $O_{i,j}$ can be processed by $M_{i,j}$ unrelated machines where machine e that processes $O_{i,j}$ can be donated by $E_{i,j,e}$. We assume that the start processing time and unit processing time of stage $O_{i,j}$, which depend on the assigned machine $E_{i,j,e}$ in sub-lot $B_{i,j,k}$, are denoted by $S_{i,j,k,e}$ and $UP_{i,j,k,e}$ respectively. If the job J_i can skip the stage j, then we will assume $O_{i,j}$ to be $\emptyset$ and set $UP_{i,j,k,e}$ to be 0. Each job $J_i \in \{J_1, J_2, \dots, J_n\}$ has a time window (R_i, D_i) where R_i, namely release date, is the earliest time that the job can begin and D_i is the due time of job i. The notations including parameters, indices and variables used in the mathematical model are shown in Table 1.

Table 1 Parameters, indices and variables used in the models.

Notations	Description
n	number of jobs
m	number of maximum stages for all jobs
M	a large positive constant
$L_{i,j}$	set of sub-lots separated by stage j of job J_i
$M_{i,j}$	set of machines that can process stage j of job J_i
J_i	index of job i
$O_{i,j}$	index of stage j in job J_i
$B_{i,j,k}$	index of batch k of stage $O_{i,j}$
$E_{i,j,e}$	index of machine e that process $O_{i,j}$
R_i	the release time of job J_i
D_i	the due time of job J_i
Q_i	the quantity of job J_i
$Set_{i1,\ i2}$	the setup time between jobs $i1$ and $i2$
$Q_{i,j,k}$	the quantity of sub-lot $B_{i,j,k}$
$S_{i,j,k,e}$	the start time of sub-lot $B_{i,j,k}$ on machine $E_{i,j,e}$
$UP_{i,j,k,e}$	the unit process time of sub-lot $B_{i,j,k}$ on machine $E_{i,j,e}$
C_i	the end time of job J_i
$X_{i1,j1,k1,e,i2,j2,k2,e}$	Takes value 1 if sub-lot $B_{i2,j2,k2}$ is processed right after sub-lot $B_{i1,j1,k1}$ on machine $E_{i,j,e}$ and 0 otherwise
$Y_{i,j,k,e}$	Takes value 1 if sub-lot $B_{i,j,k}$ is processed on machine $E_{i,j,e}$ and 0 otherwise

The objective of the scheduling problem is to minimize the makespan, which is shown in the objective function.

$$F = Min(Max(C_i)) \quad \forall i \in n \tag{1}$$

In the objective function, the makespan is defined as the complete time of all the jobs and is formulated as follows.

$$C_i \geq S_{i,j,k,e} + Set_{i1,i} + UP_{i,j,k,e} \times Q_{i,j,k} \quad \begin{matrix} \forall i, i1 \in n, j = j1 = m, \forall k \in L_{i,j}, \forall k1 \in L_{i1,j1}, \\ \forall e \in E_{i,j}, i1 = \{i1 | X_{i1,j1,k1,e,i,j,k,e} = 1\} \end{matrix} \tag{2}$$

To ensure a feasible schedule, constraints need to be fulfilled. Constraint 3 ensures all the sub-lots will be arranged when we complete the schedule.

$$\sum_{k \in L_{i,j}} Q_{i,j,k} = Q_i \quad \forall i \in n, \forall j \in m \tag{3}$$

Constraint 4 ensures each sub-lot should be processed by exactly one machine.

$$\sum_{e \in E_{i,j}} \sum_{k2 \in L_{i2,j2} \setminus \{k1\}} x_{i1,j1,k1,e,i2,j2,k2,e} = 1 \quad \forall i1, i2 \in n, \forall j1, j2 \in m, \forall k1 \in L_{i1,j1} \tag{4}$$

Constraint 5 guarantees all the sub-lots at the same stage will be processed on the same machine.

$$y_{i,j,k1,e1} = 1 \quad \forall i \in n, \forall j \in m, \forall k1 \in L_{i,j}, e1 \in \{e \mid Y_{i,j,k1,e} = 1\} \tag{5}$$

Constraint 6 indicates that one machine can only process one sub-lot at a time.

$$\sum_{k2 \in L_{i2,j2}} x_{i1,j1,k1,e,i2,j2,k2,e} = 1 \quad \forall i1, i2 \in n, \forall j1 = j2 \in m, \forall k1 \in L_{i1,j1}, \forall e \in E_{i,j} \tag{6}$$

Constraint 7 is introduced to ensure the start time of any two consecutive sub-lots $B_{i1,j1,k1}$ and $B_{i2,j2,k2}$ on the same machine to be strictly increasing. The term ensures the constraint applies only if sub-lot $B_{i2,j2,k2}$ is processed right after sub-lot $B_{i1,j1,k1}$ on machine e.

$$\begin{matrix} S_{i1,j1,k1} + Set_{i1,i2} + UP_{i1,j1,k1,e} \times Q_{i1,j1,k1} - S_{i2,j2,k2} \leq M(1 - x_{i1,j1,k1,e,i2,j2,k2,e}) \\ \forall i1, i2 \in n, \forall j1 = j2 \in m, \forall k1 \in L_{i1,j1}, \forall k2 \in L_{i2,j2}, \forall e \in E_{i,j} \end{matrix} \tag{7}$$

Constraint 8 represents a sub-lot quantity constraint. At any given time t, the finished quantity of all the batches at stage $O_{i,j}$ should not excess the finished quantity of the previous stage $O_{i,j-1}$.

$$\begin{aligned}
&\sum_{k1\in\Phi(k1)} Q_{i,j-1,k1} \geq \sum_{k2\in\Phi(k2)} Q_{i,j,k2} \\
&\Phi(k1) \in \{k \mid S_{i,j-1,k} + Set_{i1,i} + UP_{i,j-1,k,e1} \times Q_{i,j-1,k} \leq t\}, \\
&\Phi(k2) \in \{k \mid S_{i,j,k} + Set_{i2,i} + UP_{i,j,k,e2} \times Q_{i,j,k} \leq t\}, \\
&\forall i, i1, i2 \in n, \forall j \in m, \forall k \in L_{i,j}, \forall k1 \in L_{i1,j1}, \forall k2 \in L_{i2,j2}, \\
&i1 = \{i1 | X_{i1,j,k1,e1,i,j,k,e1} = 1\}, i2 = \{i2 | X_{i2,j,k2,e2,i,j,k,e2} = 1\}
\end{aligned} \tag{8}$$

The start time for each job cannot be earlier than the ready time for the job, and this is satisfied by constraint 9.

$$S_{i,j,k,e} \geq R_i \quad \forall i \in n, \forall j \in m, \forall k \in L_{i,j} \tag{9}$$

4 Two-stage ant colony algorithm with lot sizing

Due to the high complexity of the hybrid flow shop scheduling problem with lot sizing and calendar constraints, the exact method cannot obtain a feasible solution in an acceptable computational time. Although many heuristic algorithms based on dispatching rules have been employed with higher efficiency, the quality of the solution often deteriorates as the problem size increases because of the existence of many local optima. An intelligent algorithm has obvious advantages in solving complex production scheduling problems due to its optimization goal of seeking satisfactory solutions, easy integration of problem knowledge and expert experience, and procedure robustness.

As a promising swarm intelligence algorithm, the ant colony algorithm (ACA) has been widely used to solve large-scale combinatorial optimization problems since its first introduction by Dorigo (1992). The advantages are mainly due to its positive feedback, concurrency, robustness, global search, and independent of strict mathematical properties, among others. As an algorithm aiming to search for an optimal path in a graph, ACA was first introduced to solve the travel sales problem (TSP) successfully (Dorigo et al. 1997; Dorigo and Gambardella. 1997). Since then, many studies have emerged on the implementation of ACA to solve the combinatorial problem, especially scheduling problem. Now many scholars are working on solving the hybrid flow shop scheduling problem by using ACA, which is similar to the PCB

scheduling problem. Ying and Lin (2006) proposed a novel ant colony system heuristic to solve multistage hybrid flow shop scheduling problem with multiprocessor tasks, verified the superiority of ACA for solving these problems with the comparison of genetic algorithms and simulated annealing algorithms. Khalouli et al. (2011) decomposed a multistage hybrid flow shop scheduling problem into assignment and sequencing sub-problems, and proposed ant colony optimization algorithm to minimize the makespan. Qin et al. (2015) proposed an ACA based rescheduling approach for a dynamic hybrid flow shop scheduling problem with uncertain processing times.

The hierarchical approach can decompose a complex problem into several small problems that are easy to be solved, and thus this approach reduces the difficulty of solving the problem (Arnaout et al. 2010). Therefore, the hierarchical approach has broad prospects for such extremely complicated problems as the PCB scheduling problem. Besides, traditional scheduling problems rarely consider batching of items, and therefore scheduling individual item is not available in actual production processes. But in practice, each job will contain multiple items, and batching will enhance production tempo and improve equipment utilization (Wang et al. 2019). In this paper, we develop a solution procedure based on ACA for hybrid flow shop scheduling with lot sizing and calendar constraints.

4.1 Two stage structure and algorithm flow chart

Considering the complexity of the investigated problem, we decompose the original problem into two highly coupled sub-problems, which are job sequencing and lot scheduling of a job. This means that a job sequence needs to be determined firstly and then performs lot scheduling of each job till the time that all jobs are finished. This method is similar to the one proposed in Arnaout et al. (2010). As described above, ACA has been proved to be a prominent tool for scheduling problems. We adopt ACA to solve the sub-problems and propose a two-stage ant colony algorithm with lot sizing (TSACAWLS). The flow chart of proposed algorithm is shown in Fig.3, and the structure of the proposed two-stage ACA is shown in Fig.4. As shown

in the figure, the investigated problem is divided into two highly coupled stages: at stage 1, we determine the sequence of jobs; and at stage 2, we separate the production stages of each job into multiple sub-lots and assign all sub-lots to the machines. We decide to manage each of these stages with an ant system scheme.

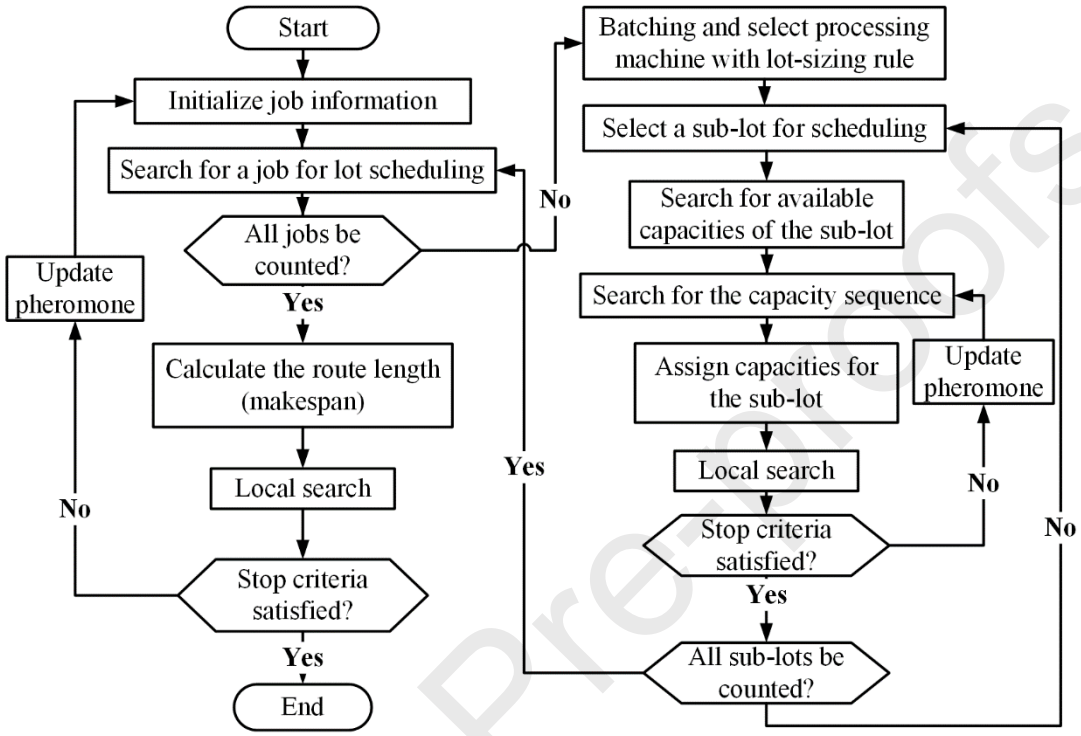


Fig.3 Flow chart of two-stage ant colony algorithm with lot sizing

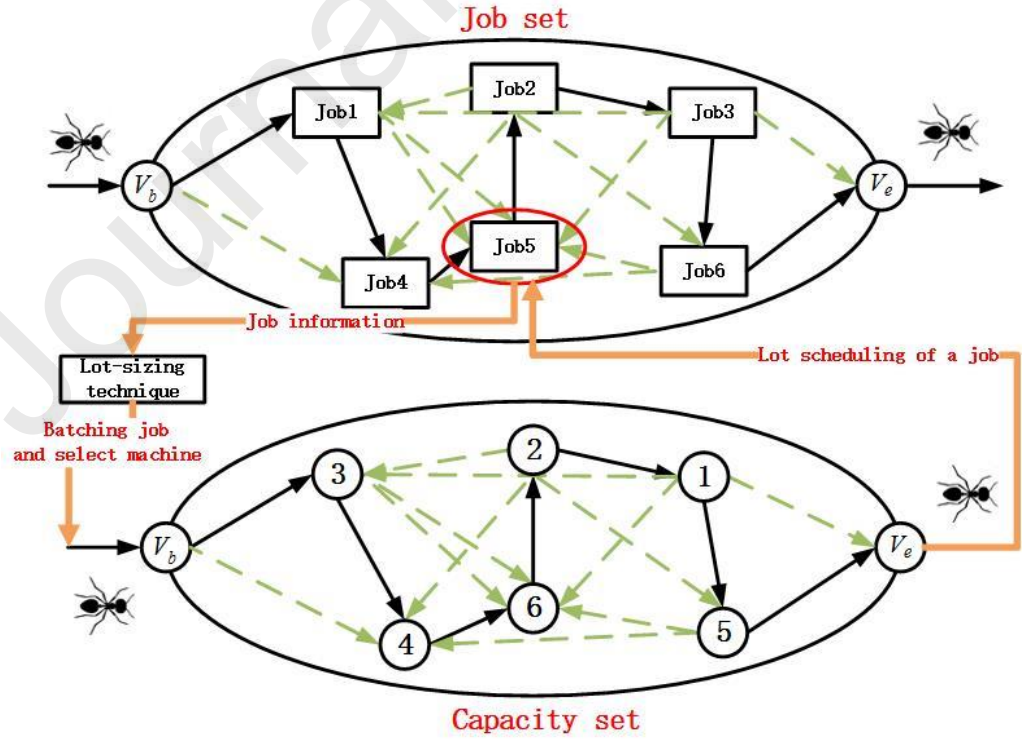


Fig.4 Structure of two-stage ant colony algorithm with lot sizing

4.2 Stage 1: Job sequencing

Stage 1 mainly deals with job sequencing problem to minimize the makespan of all jobs. We express the output of first stage with a vector (S_1) that contains n entries and each entry represents a job. We arrange the sub-lots of jobs to machines according to the sequence in the vector. For instance, if we have 10 jobs to be scheduled, the following vector: S_1 = [3 1 5 6 2 9 7 8 4 10] will represent the sequence arrangement of all the jobs. We introduce $\tau_{i,j}^{I}$ and $\eta_{i,j}^{I}$ to express the pheromone trail and the visibility of ant respectively. The possibility to select job j after i and the value of $\eta_{i,j}^{I}$ is calculated as follows:

$$P_{ij}^{I} = \frac{(\tau_{ij}^{I})^{\alpha} \cdot (\eta_{ij}^{I})^{\beta}}{\sum_{l \in \Phi} (\tau_{il}^{I})^{\alpha} \cdot (\eta_{il}^{I})^{\beta}} \tag{10}$$

$$\eta_{ij}^{I} = 1/Set_{i,j} \tag{11}$$

After all ants finish their paths, we update the pheromone amounts in each link locally by reducing the amount due to evaporation and globally by increasing the amounts of pheromone in the routes constructed by the ant that produces the best objective function (OF^{best}). This is estimated according to the following formulas:

$$\tau_{ij}^{I} \leftarrow (1-\rho)\tau_{ij}^{I} + \phi \cdot \Delta\tau_{i,j}^{I,best} \tag{12}$$

$$\Delta\tau_{i,j}^{I,best} = \begin{cases} 1/OF^{best} & if\ arc(i,j)\ is\ used\ by\ best\ ant \\ 0 & otherwise \end{cases} \tag{13}$$

Where OF^{best}, representing the makespan of the best job sequence, is the objective function with respect to the best vector S_1^{best}. The objective function of a job sequence can be calculated through stage 2.

4.3 Stage 2: Lot scheduling and lot sizing

In order to calculate the objective function of a job sequence given by stage 1, we would schedule the sub-lots at stage 2 after separating the production stages of the job

into several sub-lots. The model at stage 2 can be separated into two highly coupled sub-problems: sub-lots scheduling problem and lot sizing problem. For the first problem, we solve by using another ant system scheme for each production stage. For the second problem, we propose a lot sizing technique to separate each stage of a job into several sub-lots in the process of scheduling.

As we emphasize previously, the machines at each stage are capacitated by calendar, hence they are not always available in the investigated hybrid flow shop. The calendar sets the work shifts of all machines. A work shift, also called a capacity, is a segment of continuous available time of a machine. We schedule each stage of a job with an ant system scheme and express the output with a vector (S_2) that contains m entries with each entry representing a capacity. After that, a sub-lot will be arranged to the capacity according to the sequence in the vector. For instance, if we have 15 capacities for a job to be sequenced, then the following vector: S_2 = [5 2 1 12 13 4 11 14 8 3 7 9 10 15 6] will represent the arrangement sequence of capacities for the sub-lots of the job at that production stage. In other words, we will assign the capacity 5 as the highest priority to arrange the job. As we describe previously, several machines will be capable of processing a production stage $O_{i,j}$, it is vital to choose a suitable machine to process the job at the production stage. We assume the set of machines at stage $O_{i,j}$ to be $E_{i,j}$ and entitle each machine in $E_{i,j}$ with an objective value OV. The machine with the largest OV will be selected for the production stage.

$$OV = N / \sum_{c=1}^{N} \sum_{l \in L_{i,j-1}} \alpha_l T_c \quad \forall i \in n, \forall j \in m \tag{14}$$

$$\alpha_l = \frac{Q_i}{\sum_{k=1}^{l} Q_{i,j-1,k}} \quad \forall i \in n, \forall j \in m \tag{15}$$

$$T_c = \begin{cases} S_{i,j}(c) - Max(C_{i,j-1,k}) & if\ S_{i,j}(c) \geq Max(C_{i,j-1,k}) \\ 0 & else \end{cases} \tag{16}$$

$$\forall i \in n, \forall j \in m, \forall k \in L_{i,j-1}$$

Where α_l is the weight of quantity of the sub-lot.

T_c is the weight of start time.

N is the available number of capacity which has a start time later than the earliest end time of the previous stage and end time earlier than the due date.

$S_{i,j}(c)$ is the start time of capacity c.

The pheromone trail ($\tau_{i,j}^{II}$) is defined at this stage to indicate the latest completion time of the production stage. In addition to the pheromone, we introduce $\eta_{i,j}^{II}$ to express the visibility of ant. The possibility to select capacity j after i and the value of $\eta_{i,j}^{II}$ are calculated as follows:

$$P_{ij}^{II} = \frac{(\tau_{ij}^{II})^{\alpha} \cdot (\eta_{ij}^{II})^{\beta}}{\sum_{l \in \Phi} (\tau_{il}^{II})^{\alpha} \cdot (\eta_{il}^{II})^{\beta}} \tag{17}$$

$$\eta_{ij}^{II} = 1/End_j \tag{18}$$

Where End_j is the end time of capacity j

After all ants finish their paths, we will update the pheromone locally and globally with the similar manner of AS scheme at stage 1.

$$\tau_{ij}^{II} \leftarrow (1-\rho)\tau_{ij}^{II} + \phi \cdot \Delta\tau_{i,j}^{II,best} \tag{19}$$

$$\Delta\tau_{i,j}^{II,best} = \begin{cases} 1/OV^{best} & if\ arc\ (i,j)\ is\ used\ by\ best\ ant \\ 0 & otherwise \end{cases} \tag{20}$$

Where OV^{best} is the objective value of the best vector at stage 2, representing the minimum completion time of the stage. It can be calculated by the vector S_2 combined with the lot sizing technique below. Algorithm 1 shows the pseudo code for stage 2.

Algorithm 1: Lot Scheduling

1: Set $i = 1$
2: **while** $i \leq n$ **do**
3: Select J_i in vector S_1
4: Set $j = 1$
5: **while** $j \leq m$ **do**
6: Schedule $O_{i,j}$ based on the lot sizing procedure
7: $j \Leftarrow j + 1$
8: **if** $O_{i,j} = \emptyset$ **then**
9: $j \Leftarrow j + 1$
10: **end if**
11: **end while**

12: **end while**

In order to separate the stage into multiple sub-lots and calculate the completion time of the production stage, we propose a lot sizing method. Algorithm 2 shows the pseudo code of lot sizing procedure for $O_{i,j}$.

Algorithm 2: Lot sizing procedure for $O_{i,j}$

1: $E_{i,j,e}, S_2 \Leftarrow O_{i,j}$ and $N \Leftarrow$ number of entries in S_2
2: Set $k = 1$ and $l=1$
3: **while** $k \leq N$ **do**
4: Calculate $DQ_{i,j,k} = (C_{i,j,k,e} - S_{i,j,k,e}) * eff_e$
5: **if** $DQ_{i,j,k} \geq Q_i$ **then**
6: $Q_{i,j,l} = Q_i$, $DQ_{i,j,k} = DQ_{i,j,k} - Q_i$ and **stop**
7: **else**
8: Calculate $OQ_{i,j,k} = \sum_{k_1 \in \Phi(k_1)} Q_{i,j-1,k_1} - \sum_{k_2 < k} Q_{i,j,k_2}$, $\Phi(k_1) = \{k_1 | C_{i,j-1,k_1} < S_k\}$
9: **if** $DQ_{i,j,k} \geq OQ_{i,j,k}$ **then**
10: $Q_{i,j,l} = OQ_{i,j,k}$ and **stop**
11: **else**
12: $Q_{i,j,l} = DQ_{i,j,k}$, $Q_i = Q_i - DQ_{i,j,k}$ and $l = l + 1$
13: **end if**
14: **end if**
15: $k \Leftarrow k + 1$
16: **end while**

4.4 Local search strategy

It is known that ACA usually provides very competitive solutions when integrated with local search. Therefore, we include a local search algorithm in our implementation of ACA. After the ant k finishes its route search process, we generate neighbouring solutions for the ant in S_1 and S_2. If the local search generates a better solution at that iteration, we will use the local search solution at that iteration to update the pheromone. The neighbouring solution for S_1 is generated by rotating a random entry in the vector to the first position while that for S_2 by swapping two randomly generated entries in the vector. Algorithm 3 shows the pseudo code for the local search.

Algorithm 3: Local Search

1: Set *LocalIteration* = 1
2: **while** *LocalIteration* ≤ *MaxNoLocalIteration* **do**
3: Generate $rv \Leftarrow$ U (0,1)

4: **if** $rv < 0.5$ **then**
5: Generate S_1 (S_2)
6: Calculate OV
7: **if** $OV(LocalIteration) < OV(Ant)$ **then**
8: $OV(Ant) = OV(LocalIteration)$
9: **end if**
10: **end if**
11: **end while**

The innovative aspect of our solution methodology is the use of two highly coupled hierarchical trails to solve the problem: one for sequencing the jobs, and the other is combined with lot sizing technique for lot scheduling of a job. The pseudo code for proposed algorithm is shown in Algorithm 4.

Algorithm 4: Complete Algorithm

1: Populate the path with specified pheromone amounts ($\tau_{i,j}^{I}, \tau_{i,j}^{II}$)
2: Set *Iteration* = 1
3: while *Iteration* ≤ *MaxNoIteration*
4: Set *ant*= 1
5: **while** *ant* ≤ *MaxNoAnt* **do**
6: Search for S_1 in stage 1, refer to Eqs. 10, 11, 12 and 13
7: Schedule all the entries in S_1 with Algorithm 1
8: Algorithm 3 in stage 2
9: Update pheromone amounts locally and globally in stage 2
10: *ant= ant+1*
11: **end while**
12: Algorithm 3 in stage 1
13: Update pheromone amounts locally and globally in stage 1
14: *Iteration = Iteration +1*
15: **end while**

5 Results

To investigate the effectiveness of the proposed two-stage ant colony algorithm for hybrid flow shop scheduling with lot sizing and calendar constraints in printed circuit board assembly, uniform experiments are utilized to determine the appropriate parameter values for the ACA that will minimize the makespan of all the jobs. We verify the effectiveness of the algorithms with real data from the PCB assembly enterprise. The data can be classified into two categories.

1) Job information. 32 jobs are generated for the test. A job contains at most 5 stages, namely, surface mounting, reflow soldering, automatic/manual insertion, wave soldering and Test, marked stage 1-5 respectively. The job information tells us the release time, due date and quantity for each job, as shown in Table 2(a).

2) Machine information. As described above, stage 1 and 2 represent the front and back side of surface mounting, PCBs with one side mounting can skip stage 2 and PCBs without large elements will skip stage 3 (Plug-in). Table 2(b) shows the work shift and process time for all machines. As can be seen from the table, machines 1 and 2 are bottleneck machines.

The experiments are implemented in Microsoft Visual Studio 2008 in C# language and tested on Window 7 with a Core2 E8400 CPU at 2.8 GHz and 2GB of RAM.

Table 2 Real data from a PCB assembly enterprise

a) Job information

JobNo	Quantity	Release Time	Due Date
1	900	2017/5/24 8:00	2017-6-3 0:00:00
2	800	2017/5/24 8:00	2017-6-3 0:00:00
3	600	2017/5/24 8:00	2017-6-4 0:00:00
4	900	2017/5/24 8:00	2017-6-2 0:00:00
5	800	2017/5/24 8:00	2017-6-3 0:00:00
6	800	2017/5/24 8:00	2017-6-3 0:00:00
7	700	2017/5/24 8:00	2017-6-4 0:00:00
8	800	2017/5/24 8:00	2017-6-1 0:00:00
9	600	2017/5/24 8:00	2017-6-3 0:00:00
10	700	2017/5/24 8:00	2017-6-1 0:00:00
11	800	2017/5/24 8:00	2017-6-4 0:00:00
12	600	2017/5/24 8:00	2017-6-1 0:00:00
13	600	2017/5/24 8:00	2017-6-1 0:00:00
14	800	2017/5/24 8:00	2017-6-1 0:00:00
15	700	2017/5/24 8:00	2017-6-3 0:00:00
16	600	2017/5/24 8:00	2017-6-4 0:00:00
17	700	2017/5/24 8:00	2017-6-1 0:00:00
18	700	2017/5/24 8:00	2017-6-4 0:00:00
19	900	2017/5/24 8:00	2017-6-1 0:00:00
20	800	2017/5/24 8:00	2017-6-3 0:00:00
21	900	2017/5/24 8:00	2017-6-4 0:00:00

22	900	2017/5/24 8:00	2017-6-2 0:00:00
23	700	2017/5/24 8:00	2017-6-3 0:00:00
24	900	2017/5/24 8:00	2017-6-3 0:00:00
25	800	2017/5/24 8:00	2017-6-4 0:00:00
26	800	2017/5/24 8:00	2017-6-2 0:00:00
27	900	2017/5/24 8:00	2017-6-4 0:00:00
28	900	2017/5/24 8:00	2017-6-2 0:00:00
29	800	2017/5/24 8:00	2017-6-2 0:00:00
30	800	2017/5/24 8:00	2017-6-3 0:00:00
31	700	2017/5/24 8:00	2017-6-4 0:00:00
32	700	2017/5/24 8:00	2017-6-2 0:00:00

b) Machine information

Stage	Machine	Shift mode	ST	ET	PT (PCB/h)	ST	ET
1(2)	1	2 shifts	8:00	20:00	U(30,40)	20:00	8:00(next day)
	2	2 shifts	8:00	20:00	U(40,50)	20:00	8:00(next day)
3	3	1 shift	8:00	20:00	U(80,100)	-	-
	4	1 shift	8:00	20:00	U(80,120)	-	-
4	5	1 shift	8:00	20:00	U(80,100)	-	-
	6	1 shift	8:00	20:00	U(80,120)	-	-
5	7	1 shift	8:00	20:00	U(80,100)	-	-
	8	1 shift	8:00	20:00	U(100,120)	-	-
	9	1 shift	8:00	20:00	U(80,120)	-	-

* ST: Start Time; ET: End Time; PT: Process Time

5.1 Parameters setting

The values of investigation factors considered in this experiment have been set as four different levels, respectively as follows: α: (1,2,3,4), β: (2,3,4,5) and ρ: (0.1,0.2,0.3,0.4); where ρ is the pheromone evaporation. The choice of these values is based on many preliminary runs under different settings. To reduce the number of runs but reach sound conclusions, uniform experiments are utilized, which have shown to be an effective design (Fang et al. 2018). Table 3 shows the value combination of the factors. We donate the 16 groups of parameters in this table to be Com1, Com2, …, Com16 respectively.

Table 3 Design of uniform experiment

Com	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
α	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4

β	2	4	3	5	5	3	4	2	3	5	2	4	4	2	5	3
ρ	0.3	0.1	0.2	0.4	0.2	0.4	0.3	0.1	0.1	0.3	0.4	0.2	0.4	0.2	0.1	0.3

We select jobs 1-10 in Table 1(a), and run 20 times for each group of parameters. The value $DEV = \frac{x - \min}{\max - \min}$ is defined to evaluate the results, where x is the average value of makespan for each group of parameters, *min* is the global best value and *max* is the global worst value in all the experiments. The results with lower *DEV* are better. Finally, we sequence the *DEVs* according to their value and mark the 95% confident interval of every Com. The results are shown in Fig.5. It can be clearly seen the Com 11 makes the best performance.

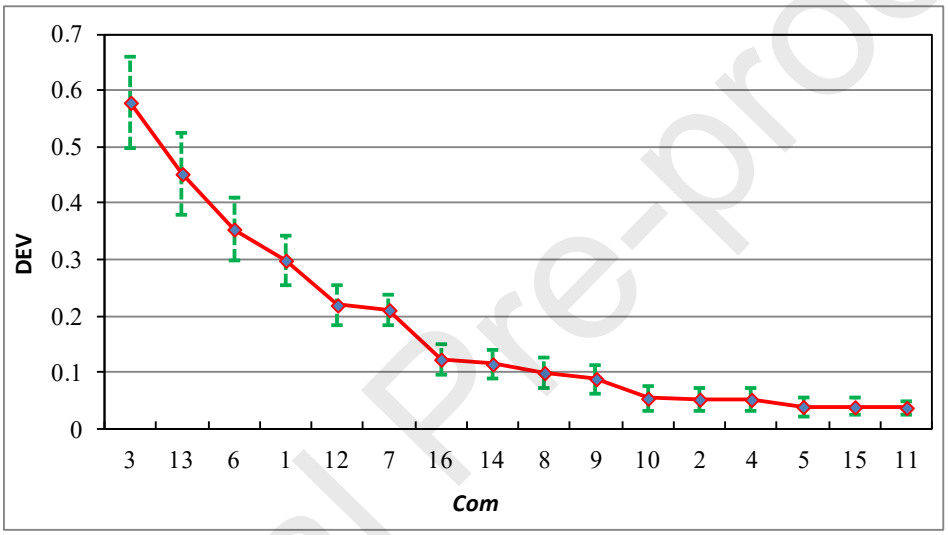


Fig.5 Sorted results of the Coms according to their *DEVs* value

In order to analyse the reason of the significant difference, we adopt the Post Hoc analysis to the results. Table 4 shows the comparison of Com11 to other Coms. Though Com11 and Com15, 5, 4, 2, 10, 9 and 8 do not exist significant difference at the confident interval of 0.05, the mean value of Com11 is lower than the other Coms. In order to improve the chance of obtaining the optimal result, we adopt the parameters in Com 11.

Table 4 Post Hoc Analysis on the comparison of Com11 to other Coms

Com (I)	Com (J)	Mean Difference (I-J)	Std. Error	Sig.	95% Confident Interval	
					Lower	Upper
11	3	-0.54135*	0.05455	0.000	-0.69773	-0.38497
	13	-0.41472*	0.05455	0.000	-0.55594	-0.27350

6	-0.31667*	0.05455	0.000	-0.42621	-0.20713
1	-0.26127*	0.05455	0.000	-0.29134	-0.23120
12	-0.18192*	0.05455	0.019	-0.20684	-0.15699
7	-0.17333*	0.05455	0.030	-0.19325	-0.15341
16	-0.08582*	0.05455	0.038	-0.09485	-0.07679
14	-0.07737*	0.05455	0.043	-0.09540	-0.05934
8	-0.06192	0.05455	0.071	-0.08012	-0.04371
9	-0.05128	0.05455	0.196	-0.06917	-0.03339
10	-0.01668	0.05455	0.235	-0.02561	-0.00776
2	-0.01489	0.05455	0.542	-0.02332	-0.00645
4	-0.01457	0.05455	0.683	-0.02357	-0.00558
5	-0.00183	0.05455	0.729	-0.00953	0.00587
15	-0.00140	0.05455	0.836	-0.00598	0.00319

5.2 Convergence validation

As it is difficult to theoretically prove the convergence of the algorithm, we validate the convergence with real examples. We run the algorithm with jobs 1-10 in table1 (a) three times with the parameters in Com11. The other parameters are set as follows: *Ant* = 5, *Iteration* = {3, 6, 9, …, 99}. The ACA at stage 1 and stage 2 followed the same parameters setting. Fig.6 shows the best results with respect to every iteration. It can be seen that the algorithm would converge to an optimal value with the iteration to be around 66.

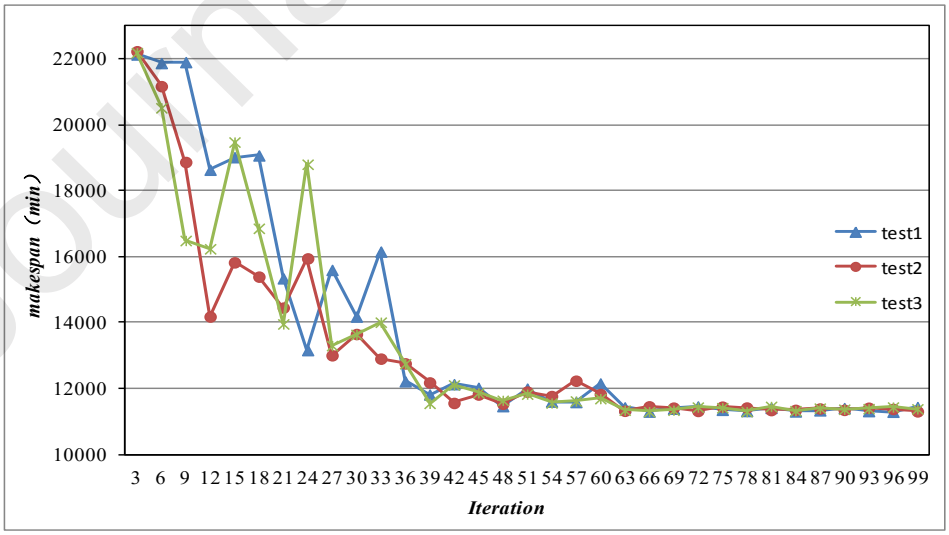


Fig.6 Verify the convergence of the algorithm

5.3 Comparisons with other heuristics

As mentioned above, few works have dealt with hybrid flow shop scheduling problem with lot sizing and calendar constraints. For lack of benchmark instances and comparison algorithms, we will illustrate the superiority of ACA and lot sizing respectively. We compare our purposed two-stage ant colony algorithm with lot sizing (TSACAWLS) with two-stage Simulated Annealing with lot sizing (TSSAWLS) in order to validate the superiority of the ACA algorithm under lot sizing and make a comparison with two-stage ant colony algorithm (TSACA) to verify the effectiveness of the lot-sizing technique. The above three algorithms are compared with two-stage earliest due date (TSEDD) rule to check that the meta-heuristic algorithms outperform the dispatch rule.

TSSAWLS shares the same structure with the TSACAWLS and is formed by replacing the ACA algorithms in TSACAWLS with SA algorithm proposed by Marimuthu (2007). As the SA algorithm is robust, we adopt the same parameters in that study. TSACA is proposed by removing the lot sizing technique in TSACAWLS, and TSEDD is constructed by replacing the ACA algorithm in TSACA with EDD rule. The algorithms are run on different sizes of problem (3, 6, 10, 14, 18, 24, 28 and 32 jobs) and each size is tested with 20 instances of the problem. The jobs are selected from Table 1.

Table 5 Solutions for different algorithms

Jobs	TSACAWLS		TSSAWLS		TSACA		TSEDD	
	avg	*min*	*avg*	*min*	*avg*	*min*	*avg*	*min*
3	4553	4553	4762	4762	5573	5573	7766	7766
6	7997.1	7820	9261.2	8669	9879.3	9356	11211	11211
10	12076.8	11376	13066.4	11986	15378.5	14832	18775	18775
14	15063.7	14001	15898.4	14532	20498.8	19768	23572	23572
18	17938.3	16871	19179.1	17638	26386.7	25673	29714	29714
24	25187.5	24144	26663.6	24873	30465.2	29659	37933	37933
28	27998.4	26950	29901.1	27960	35658.3	34786	43785	43785
32	32174.3	30563	34832.9	31835	41513.6	40365	54000	54000

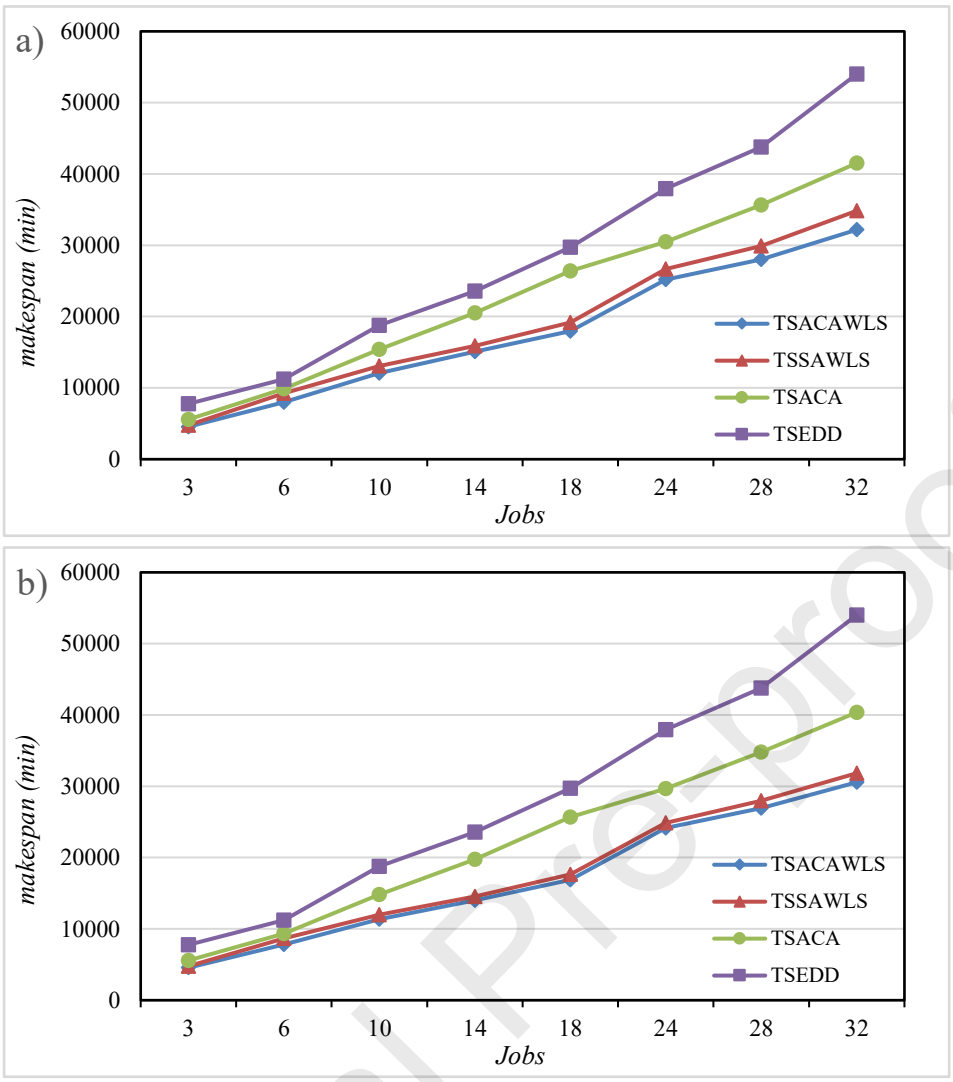


Fig.7 a) Average value for different algorithms, and b) Optimal value for different algorithms

Table 5, and Fig.7 show the average value (*avg*) and optimal value (*min*) for all problem structures and algorithms. It can be seen that the meta-heuristic algorithms (TSACAWLS, TSACA and TSSAWLS) outperform the dispatch rule (TSEDD), and TSACAWLS is better than TSSAWLS and TSACA. Even though TSACAWLS outperforms TSSAWLS and TSACA in all problem sizes, the latter's solutions appear to be close to the ones of TSACAWLS. The t-test is used to exam if their optimal value (*min*) and average value (*avg*) are significant different. The comparison of the *min* of TSACAWLS and TSSAWLS, the *min* of TSACAWLS and TSACA, the *avg* of TSACAWLS and TSSAWLS, the *avg* of TSACAWLS and TSACA are marked *min*(ACAL-SA), *min*(ACAL-ACA), *avg*(ACAL-SA) and *avg*(ACAL-ACA), respectively.

Tables 6(a), (b), and (c) show the results of t-test for different problem sizes. It can be clearly seen that the *min* and *avg* value of TSACAWLS are significantly better than that of TSSAWLS and TSACA.

Table 6 Results of t-test for different problem sizes

a) Samples Statistics for different groups

group	Mean	N	Std.Deviation	Std. Error Mean
min(ACAL-ACA)	17034.75	8	9364.25	3310.76375
	22501.50	8	12288.05	4344.48198
avg(ACAL-ACA)	17873.64	8	9834.35	3476.96753
	23169.18	8	12593.38	4452.43170
*min(*ACAL-SA*)*	17034.75	8	9364.25	3310.76375
	17781.88	8	9622.18	3401.95531
*avg(*ACAL-SA*)*	17873.64	8	9834.35	3476.96753
	19195.59	8	10490.57	3708.97538

b) Samples Correlation for different groups

group	N	Correlation	Sig.
min(ACAL-ACA)	8	0.991	0
avg(ACAL-ACA)	8	0.993	0
*min(*ACAL-SA*)*	8	1.000	0
*avg(*ACAL-SA*)*	8	0.999	0

c) T-test for different groups

group	Mean	Std. Deviation	Std. Error Mean	95% Confident Interval		t	DF	sig. (2-tailed)
				Lower	Upper			
min(ACAL-ACA)	-5466.75	3270.90	1156.44	-8201.29	-2732.21	-4.727	7	0.002
avg(ACAL-ACA)	-5295.54	3066.15	1084.05	-7858.91	-2732.17	-4.885	7	0.002
*min(*ACAL-SA*)*	-747.13	318.66	112.66	-1013.53	-480.72	-6.632	7	0
*avg(*ACAL-SA*)*	-1321.95	731.35	258.57	-1933.38	-710.52	-5.113	7	0.001

Two metrics are used to compare the three meta-heuristic algorithms, namely stability and speed to obtain a near-optimal solution. We define the difference of the *avg* and *min* values to express the stability of the algorithm. Thus, the stability of one algorithm can be calculated by: $\delta = avg - min$. The second metrics can be obtained by the average of computational time to obtain a near-optimal solution.

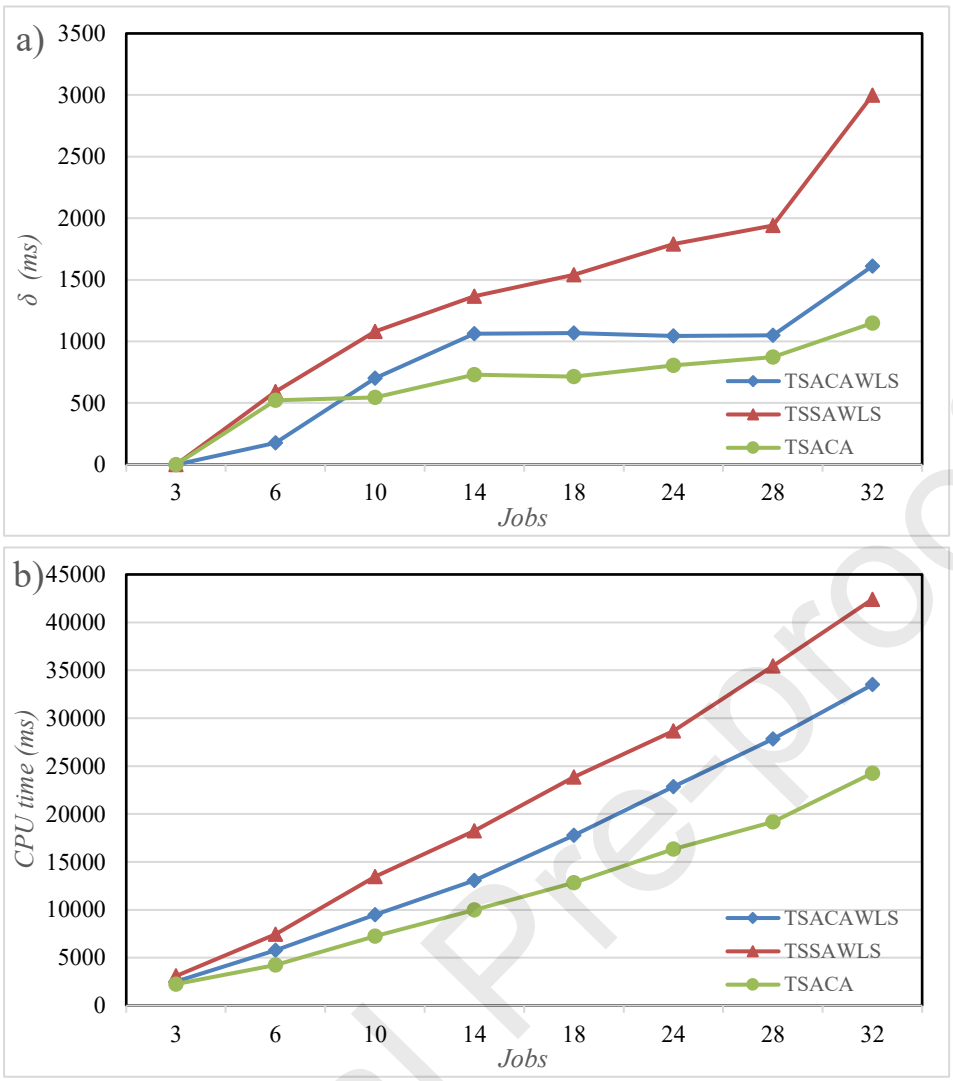


Fig.8 a) Stability of algorithms and **b)** Computational time for all algorithms

Fig.8 (a) shows the stability of the meta-heuristic algorithms. It can be seen that TSACAWLS, and TSACA outperform the TSSAWLS for all problem sizes and TSACA performs better than TSACAWLS when the number of jobs excess 11. Fig.8 (b) depicts the average computational time (CPU time) of all algorithms. TSACA and TSACAWLS require less time for a near-optimal solution than TSSAWLS, and TSACA consumes less time than TSACAWLS.

Fig.9 shows a Gantt Charts of all the algorithms for 32 jobs. It can be seen that TSACAWLS can converge to better solution than other algorithms. The algorithms with lot sizing technique (TSACAWLS and TSSAWLS) perform better than the algorithms without lot sizing (TSEDD and TSACA) in improving the utilization and balancing the load of bottleneck machines (machines 1 and 2).

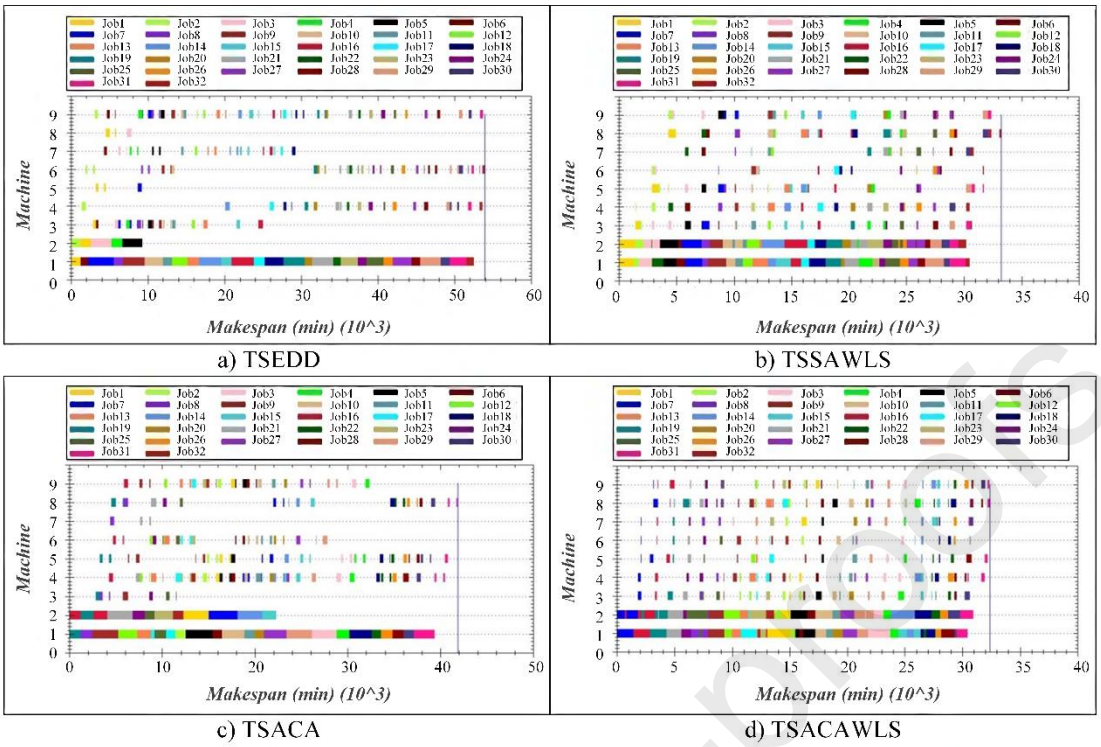


Fig.9 Gantt Charts for different algorithms

To further verify that the lot sizing can enhance the utilization and balance the load of bottleneck machines. We calculate the utilization (η) of every machine in the above Gantt charts and show the results in Fig.10. As the machines 1 and 2 are the bottleneck machines in PCB assembly. It is illustrated that the meta-heuristic methods (TSACA, TSACAWLS and TSSAWLS) can improve the utilization of bottlenecks compared to dispatch rule (TSEDD). Comparing TSACA, TSACAWLS and TSSAWLS, we can validate the conclusion that the algorithm considering lot sizing (TSSAWLS and TSACAWLS) perform better in improving the utilization and balancing the load of the bottleneck machines.

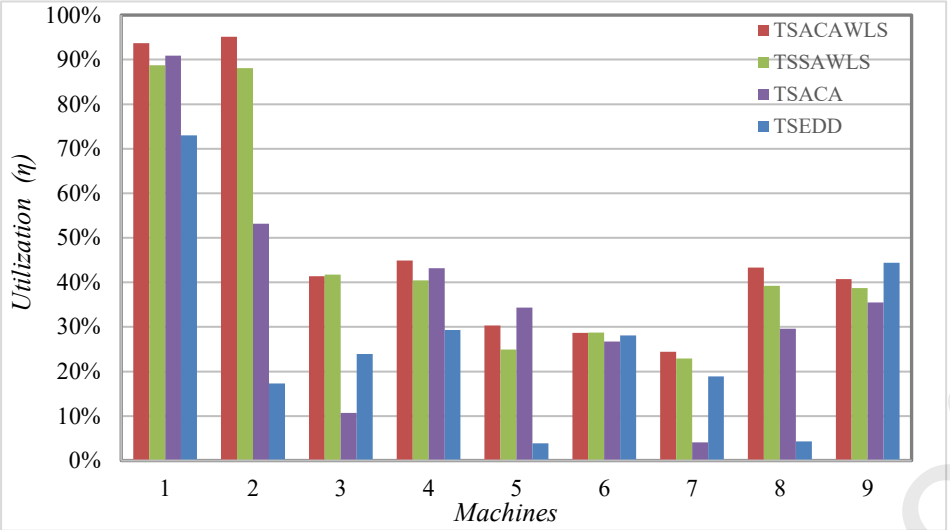


Fig.10 Result of the lines utilization using four different methods

6 Conclusions

This study deals with a multi-stage PCB scheduling problem in a semiconductor manufacturing company, which is characterized by a combination of multiple features such as lot sizing, calendar constraints, sequence-dependent setup time, unrelated parallel machines and stage skipping. A two-stage ant colony algorithm combined with the lot sizing method is proposed to solve the complicated PCB assembly shop scheduling problem by partitioning them into two sub-problems. The numerical results show that the proposed two-stage ant colony algorithm can obtain a better near-optimal solution than other approaches in terms of stability and computational time, and the lot sizing technique can further enhance the utilization and balances the load of bottleneck machines. The study on this subject makes an important contribution to the overall knowledge in the field of PCB assembly shop scheduling. From another point of view, the development of the multi-stage ACA approach also expands the application scope of ant colony optimization in the complex production scheduling field.

Although the research has dealt with several academically challenging issues, it has the following limitations and will be further studied in the future work. First, it didn't consider the group batch process after the sub-lots of PCBs complete the operation on the SMT stage, for example, in the real-life production, mounted PCBs

from different lines will be formed as a batch to be inserted and wave soldered on one or more lines. Second, it didn't involve multiple objectives, especially related with the due date such as total tardiness, which will be considered. Third, the influence of dynamic interrupts in actual production process wasn't considered, and real-time scheduling or rescheduling based on the up-to-date shop floor information will be another direction of future research.

Acknowledgements

References

Alkaya, A. F., & Duman, E. (2015). Combining and solving sequence dependent traveling salesman and quadratic assignment problems in PCB assembly. *Discrete Applied Mathematics*, *192*, 2-16.

Arnaout, J. P. , Rabadi, G. , & Musa, R. . (2010). A two-stage ant colony optimization algorithm to minimize the makespan on unrelated parallel machines with sequence-dependent setup times. *Journal of Intelligent Manufacturing, 21*(6), 693-701.

Arora, D., & Agarwal, G. (2016). Meta-heuristic approaches for flowshop scheduling problems: a review. *International Journal of Advanced Operations Management*, *8*(1), 1-16.

Bozorgirad, M. A., & Logendran, R. (2016). A comparison of local search algorithms with population-based algorithms in hybrid flow shop scheduling problems with realistic characteristics. *The International Journal of Advanced Manufacturing Technology*, *83*(5-8), 1135-1151.

Castellani, M., Otri, S., & Pham, D. T. (2019). Printed circuit board assembly time minimisation using a novel Bees Algorithm. *Computers & Industrial Engineering*, *133*, 186-194.

Chakaravarthy, G. V., Marimuthu, S., Ponnambalam, S. G., & Kanagaraj, G. (2014). Improved sheep flock heredity algorithm and artificial bee colony algorithm for scheduling m-machine flow shops lot streaming with equal size sub-lot problems. *International Journal of Production Research*, *52*(5), 1509-1527.

Chen, T. L., Cheng, C. Y., & Chou, Y. H. (2018). Multi-objective genetic algorithm for energy-efficient hybrid flow shop scheduling with lot streaming. *Annals of Operations Research*, 1-24.

Cheng, M., Mukherjee, N. J., & Sarin, S. C. (2013). A review of lot streaming. *International Journal of Production Research*, *51*(23-24), 7023-7046.

Cui, W. W., Lu, Z., Zhou, B., Li, C., & Han, X. (2016). A hybrid genetic algorithm for non-permutation flow shop scheduling problems with unavailability constraints. *International Journal of Computer Integrated Manufacturing*, *29*(9), 944-961.

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*, 342-353.

Dorigo, M. (1992). Optimization, learning and natural algorithms. PhD thesis, Politecnico di Milano, Italie.

Dorigo, M., Caro, G.D., & Gambardella, L. M. (1997). Ant colony system: A cooperative learning approach to the travelling salesman problem. *IEEE Transactions on Evolutionary Computation*, *1*, 53-66.

Dorigo, M., & Gambardella, L.M. (1997). Ant colonies for the travelling salesman problem. *Bio System, 43*, 73-81.

Fang, K., Liu, M. Q., Qin, H., & Zhou, Y. D. (2018). *Theory and application of uniform experimental designs* (Vol. 221). Springer.

González-Neira, E., Montoya-Torres, J., & Barrera, D. (2017). Flow-shop scheduling problem under uncertainties: Review and trends. *International Journal of Industrial Engineering Computations*, *8*(4), 399-426.

Han, Y., Gong, D., Jin, Y., & Pan, Q. (2017). Evolutionary multiobjective blocking lot-streaming flow shop scheduling with machine breakdowns. *IEEE transactions on cybernetics*, (99), 1-14.

Hatami, S., Ruiz, R., & Andrés-Romano, C. (2015). Heuristics and metaheuristics for the distributed assembly permutation flowshop scheduling problem with sequence dependent setup times. *International Journal of Production Economics*, *169*, 76-88.

Huang, R. H., & Yu, T. H. (2017). An effective ant colony optimization algorithm for multi-objective job-shop scheduling with equal-size lot-splitting. *Applied Soft Computing*, *57*, 642-656.

Khalouli, S., Ghedjati, F., & Hamzaoui, A. (2011). An ant colony system algorithm for the hybrid flow-shop scheduling problem. *International Journal of Applied Metaheuristic Computing (IJAMC)*, *2*(1), 29-43.

Krishnan, R. (2014). *Placement sequence optimization in pcb assembly using neighborhood search heuristics and simulated annealing*. State University of New York at Binghamton.

Lee, T., & Loong, Y. (2019). A review of scheduling problem and resolution methods in flexible flow shop. International Journal of Industrial Engineering Computations, 10(1), 67-88.

Linn, R. and Zhang, W., 1999. Hybrid flow shop scheduling: a survey. *Computers & Industrial Engineering*, 37 (1–2), 57–61.

Marimuthu, S., Ponnambalam, S. G., & Jawahar, N. (2007). Tabu search and simulated annealing algorithms for scheduling in flow shops with lot streaming. *Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture*, *221*(2), 317-331.

Meng, T., Pan, Q. K., Li, J. Q., & Sang, H. Y. (2018). An improved migrating birds optimization for an integrated lot-streaming flow shop scheduling problem. *Swarm and Evolutionary Computation*, *38*, 64-78.

Navaei, J., & ElMaraghy, H. (2017). Grouping and sequencing product variants based on setup similarity. *International Journal of Computer Integrated Manufacturing*, *30*(6), 664-676.

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. *Journal of Industrial and Production Engineering*, *33*(7), 459-471.

Noroozi, A., & Mokhtari, H. (2015). Scheduling of printed circuit board (PCB) assembly systems with heterogeneous processors using simulation-based intelligent optimization methods. *Neural Computing and Applications*, *26*(4), 857-873.

Pan, Q. K., Gao, L., Li, X. Y., & Gao, K. Z. (2017). Effective metaheuristics for scheduling a hybrid flowshop with sequence-dependent setup times. *Applied Mathematics and Computation*, *303*, 89-112.

Qin, W. , Zhang, J. , & Song, D. . (2015). An improved ant colony algorithm for dynamic hybrid flow shop scheduling with uncertain processing time. *Journal of Intelligent Manufacturing*, 1-14.

Raduly-Baka, C., Johnsson, M., & Nevalainen, O. S. (2017). Tool-feeder partitions for module assignment in PCB assembly. *Computers & Operations Research*, *78*, 108-116.

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

Seidgar, H., Zandieh, M., & Mahdavi, I. (2016). Bi-objective optimization for integrating production and preventive maintenance scheduling in two-stage assembly flow shop problem. *Journal of Industrial and Production Engineering*, *33*(6), 404-425.

Seidgar, H., Rad, S. T., & Shafaei, R. (2017). Scheduling of assembly flow shop problem and machines with random breakdowns. *International Journal of Operational Research*, *29*(2), 273-293.

Shoaardebili, N., & Fattahi, P. (2015). Multi-objective meta-heuristics to solve three-stage assembly flow shop scheduling problem with machine availability constraints. *International Journal of Production Research*, *53*(3), 944-968.

Tóth, A., Knuutila, T., & Nevalainen, O. S. (2018). Machine configuration and workload balancing of modular placement machines in multi-product PCB assembly. *International Journal of Computer Integrated Manufacturing*, *31*(9), 815-830.

Wang, S., Kurz, M., Mason, S. J., & Rashidi, E. (2019). Two-stage hybrid flow shop batching and lot streaming with variable sublots and sequence-dependent setups. *International Journal of Production Research*, 1-15.

Xu, J., Wu, C. C., Yin, Y., & Lin, W. C. (2017). An iterated local search for the multi-objective permutation flowshop scheduling problem with sequence-dependent setup times. *Applied Soft Computing*, *52*, 39-47.

Yang, H. Q. (2018). The current chinese native PCB industry like sun rising in the east —— The analysis of global Top 100 PCB companies. *Printed Circuit Information*, *26* (11), 6-11.

Ying, K. C., & Lin, S. W. (2006). Multiprocessor task scheduling in multistage hybrid flow-shops: an ant colony system approach. *International Journal of Production Research*, *44*(16), 3161-3177.

Zohali, H., Naderi, B., Mohammadi, M., & Roshanaei, V. (2019). Reformulation, linearization, and a hybrid iterated local search algorithm for economic lot-sizing and sequencing in hybrid flow shop problems. *Computers & Operations Research*, *104*, 127-138.

Acknowledgements

The authors would like to acknowledge the financial supports of the National Natural Science Foundation of China (No. 51775348, No. 51435009).