A Novel Parallel Assembly Sequence Planning Method for Complex Products Based on PSOBC

Yanfang Yang,¹ ² Miao Yang,¹ Liang Shu,³ Shasha Li,³ and Zhiping Liu¹ ²

¹School of Logistic Engineering, Wuhan University of Technology, Wuhan, China
²Engineering Research Center of Port Logistics Technology and Equipment, Ministry of Education, Wuhan, China
³The Low-Voltage Apparatus Technology Research Center of Zhejiang, Wenzhou University, Wenzhou, China

Correspondence should be addressed to Liang Shu; shuliangalbert@163.com

Received 27 December 2019; Revised 18 March 2020; Accepted 19 March 2020; Published 26 May 2020

Copyright © 2020 Yanfang Yang et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Parallel assembly sequence planning (PASP) greatly impacts on efficiency of assembly process. In traditional methods, large scale of matrix calculation still limits efficiency of PASP for complex products. A novel PASP method is proposed to address this issue. To avoid matrix calculation, the synchronized assembly Petri net (SAPN) is firstly established to describe the precedence relationships. Associated with the SAPN model, the PASP process can be implemented via particle swarm optimization based on bacterial chemotaxis (PSOBC). Characterized by an attraction-repulsion phase, PSOBC not only prevents premature convergence to a high degree, but also keeps a more rapid convergence rate than standard particle swarm optimization (PSO) algorithm. Finally, feasibility and effectiveness of the proposed method are verified via a case study. With different assembly parallelism degrees, optimization results show that assembly efficiency of the solution calculated by PSOBC method is 9.0%, 4.2%, and 3.1% better than the standard PSO process.

1. Introduction

ASP is one of the important parts of industrial manufacturing, which is able to provide fast and high efficiency guidelines for equipment assembly. High efficiency of ASP directly determines enterprise profit. However, ASP is usually difficult due to the large number of equipment components (assembly operations) and complicated priorities/constraints relations among components.

Usually ASP can be treated as an NP-hard problem [1], which is generally solved to maximize the multiple benefits during the assembly process [2]. There are several challenges remaining to construct efficient and comprehensive ASP, one of which is the modeling problem. To optimize ASP, the precedence relationships among every individual component need to be investigated. A number of methods have been developed to meet various requirements in the ASP optimization. AND/OR graph model was firstly developed by De Mello et al. [3], and it has been widely employed in assembly sequence generation researches. Precedence matrix was built and embodied into Petri net to generate the reachability graph by Yang et al. [4]. Similar assembly direction matrix, interference matrix, and sequence-relation matrix were introduced to generate optimal assembly sequence for eccentric milling machine in [5]. Such matrix method contributes to generation and acquisition of production information, and it is usually straightforward and easy to be implemented. However, due to the large number of equipment components and the complicated constrains, the size of precedence matrix is generally very large. Computational burden will limit the optimization process. To increase ASP efficiency, some other advanced methods were introduced and developed, e.g., weighted precedence graph [6], assembly tree [7], layout-based hierarchical graph [8], and part concatenation method [9–11]. In these methods, the weighted assembly graph reduces the complexity of ASP through considering qualitative and quantitative constraints. By using the assembly tree method, generation of feasible and stable sequences can be implemented through the recursive construction and stability
criteria. Part concatenation method is robust and able to address ASP problem by considering all necessary assembly predicates. However, such models are still matrix based. Not only is enormous human calculation required to investigate complex products, but also human error would occur with a large probability. To address this problem, a SAPN model is developed to describe the operations, priorities, and constraints in assembly process. Each transition is associated with the predecessors by using assembly synchronizers, which illustrates the precedence relationships. The transition status is updated in each assembly operation and the assembly can be identified by the status checking.

ASP optimization could be implemented when the assembly model is developed. Generally, ASP can be categorized into two types, the SASP and the PASP. The assembly operations of SASP are constructed in series and are easy to be implemented. Since all the operations need to follow a linear sequence, the efficiency is limited. Assembly operations of PASP can be performed in parallel and the efficiency can be increased. However, it is still challenging to assemble complex equipment with PASP. To increase efficiency and correctness, the PASP needs to be optimally designed. Moreover, optimization of PASP needs to be subjected to strict predetermined constraints.

Heuristic algorithm is an effective way to solve optimization problem with constraints. Typical methods include genetic algorithm [12–14], PSO [15, 16], ant colony optimization algorithm [17, 18], and other evolutionary algorithms [19–21]. GA is a nonlinear estimation method that generates solutions to optimization problems using techniques inspired by natural evolution [22]. To tackle ASP of complex products, a chaotic PSO approach to generate the optimal or near-optimal assembly sequences was developed in [23]. Generally, heuristic searching needs to be implemented with numerical iterations. The coding rules and heuristic rules under predetermined constraints are important to construct the efficient searching. Like what has been discussed, most of the ASP models and the coding rules are matrix based. The searching/iteration performance is limited due to the matrix calculations. We have developed the SAPN model to describe the operations, priorities, and constraints in the assembly process. Coding rules of PSO BC such as position, velocity, and diversity function are represented with index vectors instead of matrix. The consequent efficiency can be increased.

As for the heuristic rules design, the premature phenomenon is the major problem that needs to be addressed. The searching result could be attracted to the local optima if the heuristic rules are designed improperly. Inspired by the phenomenon of chemotaxis in bacteria colonies, PSO BC was proposed to improve the performance of standard PSO [24]. Characterized by an attraction-repulsion phase, PSO BC not only prevents premature convergence to a high degree, but also keeps a more rapid convergence rate than standard PSO. This algorithm has been successfully used in areas like CO2 emission problem [25] and hydropower stations operations [26]. The global and local search performance could be well balanced when the particle swarm diversity is properly controlled. Here we use a similar PSO BC framework to solve the PASP problem based on the proposed SAPN model. The diversity function is built up to avoid population premature. Feasibility and effectiveness of the proposed method are also verified via a case study. Optimization results have been compared with standard PSO method under different DPAs.

2. Establishment and Analysis of SAPN

2.1. Basic Petri Net

Definition 1 (Petri net). Let \( N = (P, T, F) \) be the basic Petri net which should satisfy the following three conditions:

1. \( P \cup T \neq \emptyset \cap P \cap T = \emptyset \),
2. \( F \subseteq P \times T \cup T \times P \),
3. \( \text{dom}(F) \cup \text{cod}(F) = P \cup T \).

Here “\( \times \)” is the Cartesian product, \( \text{dom}(F) = \{ x | \exists y: (x, y) \in F \} \), and \( \text{cod}(F) = \{ y | \exists x: (x, y) \in F \} \). Condition (3) is defined to ensure that there is no isolated place, transition, or arc in the Petri net.

2.2. Symbols in SAPN

Definition 2 (synchronizer). Let \( p = (T_1, T_2, (a_1, a_2)) \) be a synchronizer, where \( \forall t_i \in T_1, t_j \in T_2: p = t_i^* \rightarrow t_j, w(t_i, p) = a_2, w(p, t_j) = a_1, T_1 = \{ t_1, t_2, \ldots, t_{m_1} \}, T_2 = \{ t_1, t_2, \ldots, t_{m_2} \}, 1 \leq a_1 \leq m_1 = |T_1|, \text{and} 1 \leq a_2 \leq m_2 = |T_2| \).

The scheme of the defined synchronizer is shown in Figure 1. The SAPN consists of a set of synchronizers corresponding to different assembly status. The synchronizers are used to illustrate the parallel and selective relations of each assembly operation, including the corresponding precedence relationships.

Definition 3 (SAPN). Let \( \text{SAPN} = (P, T, F, W, M, PT, TM) \) be the synchronized assembly Petri net consisting of assembly synchronizers. Places, transitions, arcs, and other tuples are integrated in the net. Detailed explanation of these tuples is given as follows:

\( P = \{ p_j \}, (j = 1, 2, \ldots, m) \) represents the assembly status, and \( m \) is the amount of the places.

\( T = \{ t_i \}, (i = 1, 2, \ldots, n) \) represents the assembly operations, and \( n \) is the amount of the transitions.

\( F = \{ f_k \}, (k = 1, 2, \ldots, l) \) represents the directed arcs between transitions and places, and \( l \) is the amount of the arcs.

\( W = \{ w_k \}: F \rightarrow \{ 1, 2, 3, \ldots \} \) represents the weight functions associated with arcs. The weight functions determine the amount of the required tasks for a place to transform into another one. The weight functions should be assigned based on number of the arcs which share the same input/output place in the corresponding synchronizer. For example, there are \( m \) arcs between \( T_1 \) and \( p \) sharing the same output place \( p \) in Figure 1, so the weight functions of these arcs should be both
assigned as $m_1$. Also the weight functions of $p$’s output arcs should be both assigned as $m_2$ according to similar recursion.

$M = \{m_i\}: T \rightarrow \{0, 1, 2\}$ represents the markings associated with transitions. Here $M$ is defined to represent the transition status. In the assembly process, the marking $M$ is a piecewise constant vector and the values are dependent on the transition status based on the following rules:

$$m_i = \begin{cases} 
0, & \text{if } t_i \text{ is fired,} \\
1, & \text{if } t_i \text{ is fireable,} \\
2, & \text{if } t_i \text{ is unfireable, while it hasn’t been fired.} 
\end{cases} \tag{1}$$

$PT = \{pt_i\}: T \rightarrow \{t_i\}, \ (i_1,i_2 = 1, 2, \ldots, n)$ represents the precedence relationships among equipment components. The element $pt_i$ in $PT$ is defined to collect the preceding transitions of $t_i$. In others words, if $\exists p = (T_1, T_2, (a_1, a_2)): t_i \in T_1 \land t_i \in T_2$, then the transitions in $T_i$ are the predecessors of the transitions in $T_2$.

$TM = \{tm_i\}: T$ represents operation times of the transitions.

2.3. Construction of SAPN. This subsection introduces construction approach of SAPN through the following steps.

Step 1: Generate transitions for each necessary assembly operation.

Step 2: Assign $t_{ms}$, $ms$, and $p_{ts}$ to each corresponding transition.

Step 3: Generate the starting place, the ending place, and the places between each transition and its predecessors.

Step 4: Integrate the duplicated input places of each transition.

Step 5: Generate the directed arcs from predecessors to the corresponding places and from the places to the corresponding successors.

Step 6: Assign the weight function to each arc.

SAPN model can be constructed through the detailed steps mentioned above. Step 1 and Step 2 generate the fundamental transitions. Generation of the corresponding places is processed in Step 3 and Step 4. Since SAPN is oriented to the complex products, the elements in SAPN are unavoidably large. Therefore, the integration process in Step 4 somewhat reduces the scale of SAPN model. Step 5 and Step 6 generate and assign the arcs, which link the elements in each synchronizer.

2.4. Analysis of Assembility. A transition which is fireable must satisfy the condition that its preceding transitions have all been fired in the synchronizer. Assembility of each component can be determined from the precedence relationships and the status, i.e., $PT$ and $M$. The necessary and sufficient condition of $t_i$ being fireable is given as below:

$$m_{i_1} = 1. \tag{2}$$

Equation (2) can also be rewritten as

$$\forall t_{i_2} \in pt_{i_1}: \sum m_{i_2} = 0 \land m_{i_1} \neq 0. \tag{3}$$

It is seen from (3) that $t_i$ is fireable when its preceding transitions have all been fired; namely, $m_{i_1} = 0$. Analysis of assembility is necessary in the sequence optimization process to avoid violation of precedence relationships.

3. Principles of Sequence Optimization

In this section, PSOBC framework is developed to solve the PASP problem. The algorithm alternates between phases of attraction and repulsion. Once the diversity of population is too high, the individuals will be congregated by attraction force to explore better solution. If the diversity of population is too low, the individuals will be dispersed by repulsion force to ensure better convergence. Thus the sequence optimization can be efficiently performed.

3.1. Problem Statement. Here we use a simple example to illustrate the principles of PASP. The parallel assembly scheme is shown in Figure 2. It is assumed that there are 3 manipulators and 13 individual parts of equipment. The length of each rectangle in Figure 2 represents the assembly time of the corresponding part. Three manipulators are employed and five assembly steps are involved in the process. During the process, time represents the operation time of the whole assembly process. To improve efficiency, the goal of sequence planning is set to minimize the assembly time, time. DPA can be defined to describe the biggest number of components or parts to be assembled simultaneously, i.e., the amount of the manipulators. According to the definition, the DPA of the example in Figure 2 is 3.

To improve assembly performance, time study is considered to minimize assembly time. The objective function is given in the following formulation:

$$\min (time) = \sum_{s=1}^{s} time_s. \tag{4}$$

In (4), $time_s$ represents operation time of the subassembly $s$ and $time$ represents the whole assembly time.
3.2. Coding Rules. In this subsection, coding rules for PASP problem are proposed according to its features. The rules include particle position, particle velocity, fitness function, and diversity function. Since matrix calculation is avoided, calculation process would be in high efficiency and effectiveness.

Definition 4 (particle position). Let the particle position \( \mathbf{X}_d \) be a parallel sequence, which is expressed by a vector as below:

\[
\mathbf{X}_d = [x_{d1}, x_{d2}, \ldots, x_{dn}].
\] (5)

Since \( x_{di} \) represents subassembly index of transition \( t_i \), it is supposed to be an integer.

Definition 5 (particle velocity). Let the particle velocity \( \mathbf{V}_d \) be the updating rule for the particles, which is represented by another vector as below:

\[
\mathbf{V}_d = [v_{d1}, v_{d2}, \ldots, v_{dn}].
\] (6)

Since \( v_{di} \) represents the updating rule for transition \( t_i \) of particle \( \mathbf{X}_d \), \( v_{di} \) is supposed to be a real number.

Definition 6 (fitness function). Let the fitness function be the formulation to evaluate particle positions as expressed in Definition 5

\[
FT(\mathbf{X}) = C_t - \sum_{s=1}^{S} \left[ tme_s + \delta_p F_p(\mathbf{s}) + \delta_i F_i(\mathbf{s}) \right].
\] (7)

In (7), \( C_t \) is a constant defined to convert minimization to maximization. Assembly time is considered in the fitness function to optimize assembly efficiency, while the two penalty factors/functions are used to, respectively, investigate precedence relationships and parallel principles.

To calculate the fitness function, the first step is decoding \( \mathbf{X} \) into parallel assembly sequence based on its definition, while the second step is calculating each \( \text{time}_s, F_p(\mathbf{s}), \) and \( F_i(\mathbf{s}) \). \( \text{time}_s \) is the max operation time in subassembly. \( F_p(\mathbf{s}) \) equals the number of unfireable transitions in subassembly \( s \), i.e., the max operation time of the transitions equal 2 according to analysis of assemblability in the synchronizer (3). After calculating each \( F_p(\mathbf{s}) \), marking values of transitions in the subassembly should be set as 0, and marking values of the unfired transitions in SDPN should be set as 1 if their preceding transitions have all been fired, or the values remain equal to 2. \( F_i(\mathbf{s}) \) equals 0 if operation number of subassembly \( s \) is less than DPA, or it equals DPA minus operation number.

Definition 7 (diversity of population). Let the diversity of population be the formulation shown in (8) to evaluate the average distance between all particles.

\[
F_d(P_t) = \frac{1}{|P_t||P_t|} \sum_{d=1}^{|P_t|} \sum_{i=1}^{|P_t|} (x_{di} - \bar{X}_i)^2.
\] (8)

In (8), \( |P_t| \) is the population size, while \( \bar{X}_i \) is the mean subassembly indices of \( t_i \) in the population. Diversity function is formulated to detect premature convergence, and then attraction/repulsion phase can be implemented to improve algorithm performance.

3.3. Algorithm Optimization. Let \( \mathbf{V}_d^g \) be the velocity of particle \( d \) in generation \( g \), and \( \mathbf{X}_d^g \) be the position of particle \( d \) in generation \( g \). The PSOBC algorithm should be implemented with the following equations:

\[
\mathbf{V}_d^{g+1} = \omega \times \mathbf{V}_d^g + c_1 r_1 \times (\mathbf{B}_d - \mathbf{X}_d^g) + c_2 r_2 (\mathbf{B}_d - \mathbf{X}_d^g),
\] (9)

\[
\mathbf{X}_d^{g+1} = \mathbf{X}_d^g + \mathbf{V}_d^{g+1},
\] (10)

\[
\mathbf{V}_d^{g+1} = \omega \times \mathbf{V}_d^g - c_1 r_1 \times (\mathbf{W}_d^g - \mathbf{X}_d^g) - c_2 r_2 (\mathbf{W}_d^g - \mathbf{X}_d^g).
\] (11)

In attraction phase, (9) and (10) should be exerted to optimize solutions with the principle of approaching local/global best positions. In repulsion phase, (10) and (11) should be exerted to improve convergence with the principle of escaping from local/global worst positions. The calculating rules of (9)–(11) are proposed as below.

\[
\mathbf{X}_d \pm \mathbf{X}_d = \mathbf{V} = \left[ x_{d1} \pm x_{d1}, x_{d2} \pm x_{d2}, \ldots, x_{dn} \pm x_{dn} \right],
\] (12)

\[
k \times \mathbf{V}_d = \mathbf{V} = \left[ k v_{d1} v_{d2} \ldots v_{dn} \right],
\] (13)

\[
\mathbf{V}_d \pm \mathbf{V}_d = \mathbf{V} = \left[ v_{d1} \pm v_{d1}, v_{d2} \pm v_{d2}, \ldots, v_{dn} \pm v_{dn} \right],
\] (14)

\[
\mathbf{X}_d + \mathbf{V}_d = \mathbf{X} = \left[ x_{d1} + v_{d1}, x_{d2} + v_{d2}, \ldots, x_{dn} + v_{dn} \right].
\] (15)

Since \( x \) is an integer and \( v \) is a real number, the elements in \( \mathbf{X} \) should be both rounded to integers after calculation using (15).

4. Case Study

In this paper, a metallurgical reducer is taken as an example to verify the feasibility and the efficiency of the proposed method. The DPAs are selected as 2, 3, and 4, respectively. The explosive diagram of the reducer is shown in Figure 3, including 50 components in total. The main parts of the reducer are gears, boxes, caps, balls, shafts, lubricating rings, and eccentric sleeves.

4.1. Establishing SAPN. Based on the reducer structure, the places, transitions, arcs, and other tuples in SAPN = (\( P, T, F \)), \( (W, M, PT, TM) \) are established in Section 2.3. Structure diagram of the SAPN is built as shown in Figure 4. In the net model, \( P \) is the set of places defined to represent component status, \( p_1 \) and \( p_{49} \) are the starting and ending places. \( T \) is the set of transitions, representing the assembly operations. Definitions of the transitions in the SAPN model are given in Table 1. \( F \) is the set of arcs, representing flow relations between transition and place. \( W \) is the set of weights associated with the arcs. The weight functions are assigned based on the definition mentioned in Section 2.2.
example, weight functions of the arcs between \( p_2 \) and \( t_3 \), \( p_5 \) and \( t_5 \), and \( p_{28} \) and \( t_{28} \) are assigned as 1, while weight function of the arcs between \( p_{29} \) and \( t_{41} \) is assigned as 3. \( M \) is the set of markings associated with transitions. Precedence relationships are illustrated in \( PT \) to analyze the component assembly. \( TM \) is the set of assembly times associated with each assembly operation.

It is seen from Table 1 that large size of matrices would be generated if the matrix-based method was adopted for metallurgical reducer ASP design, which would be complex and difficult to implement. This problem can be addressed through the proposed PASP framework.

### 4.2. Population Initialization

Based on the heuristic mechanism of PSOBC, population initialization is required for the sequence optimization. The stages of population initialization can be structured as a flow chart represented in Figure 5. Detailed explanation of the initialization procedure is given in the following steps.

1. **Step 1:** Obtain the precedence relationships and the initial status of the components from \( PT \) and \( M \). Build \( IP \) (an empty set of particles position) as the initial population.
2. **Step 2:** Build a new particle position \( PDS \) (an empty set of transitions).
3. **Step 3:** Build a new set of all fireable transitions \( RTS \) based on analysis of assembly.
4. **Step 4:** Build a new assembly step \( DS \) (an empty set of transitions). If \( n_r \) (the amount of transitions in \( RTS \)) exceeds DPA, randomly push 1–DPA assembly operations into \( DS \). Otherwise, push all the operations form \( RTS \) into \( DS \). Update status and marking of the transitions in \( DS \), and push the transitions from \( DS \) into \( PDS \).
5. **Step 5:** Remove the fired transitions in \( RTS \) based on \( PT \) and \( M \).
6. **Step 6:** If \( n_r \) is positive, return to step 4. Otherwise, encode \( PDS \) into a particle position (an index vector) and push it into \( IP \); return to step 7.

### 4.3. Generation of the Optimal Sequences

The key procedure of PASP is the sequence optimization process implemented to find the optimal solution. Main stages of heuristic optimization can be structured as another flow chart represented in Figure 6. Detailed explanation of the procedure is illustrated in the following steps.

1. **Step 1:** Input the initial population from initialization process. Set phase as “attraction.”
2. **Step 2:** Update the current best and worst particle position for individuals and the whole population.
3. **Step 3:** If the phase is “attraction,” calculate the new velocity and position for each particle using (9) and (10). Otherwise, calculate using (10) and (11).
4. **Step 4:** Calculate diversity function of the population. If premature convergence occurs, set the phase as “repulsion” and return to step 6. Otherwise, return to step 5.
5. **Step 5:** Calculate diversity function of the population. If premature convergence is escaped, set the phase as “attraction” and return to step 6.
**Table 1: Detailed definitions of transitions in the SAPN.**

| $T$  | Targeted part      | No.   | $PT$ | $TM$ | $M$ | $T$  | Targeted part | No.   | $PT$ | $TM$ | $M$ |
|------|---------------------|-------|------|------|-----|------|----------------|-------|------|------|-----|
| $t_1$| Null                | Null  | Ø    | 0    | 2   | $t_16$| Lubricating ring 3 | 41    | $t_16$, $t_3$ | 1.8 | 2   |
| $t_2$| Bear 12             | 18    | $t_1$| 1.9  | 2   | $t_37$| Bear 4          | 3     | $t_1$       | 2.1 | 2   |
| $t_3$| Bear 11             | 45    | $t_1$| 1.6  | 2   | $t_28$| Bear 3          | 9     | $t_1$       | 2.1 | 2   |
| $t_4$| Eccentric sleeve 5  | 44    | $t_1$| 2.1  | 2   | $t_29$| Bear 2          | 47    | $t_{24}$    | 2.1 | 2   |
| $t_5$| Spline              | 20    | $t_1$| 1.8  | 2   | $t_30$| Lock 3          | 33    | $t_1$       | 1.2 | 2   |
| $t_6$| Eccentric sleeve 4  | 24    | $t_1$| 2.1  | 2   | $t_31$| Bear 1          | 39    | $t_{18}$    | 1.4 | 2   |
| $t_7$| Bear 10             | 15    | $t_1$| 1.3  | 2   | $t_32$| Lubricating ring 2 | 42    | $t_{26}$    | 1.5 | 2   |
| $t_8$| Bear 9              | 17    | $t_1$| 1.3  | 2   | $t_33$| Sleeve 3        | 2     | $t_{27}$    | 2.1 | 2   |
| $t_9$| Gear 7              | 23    | $t_1$| 1.3  | 2   | $t_34$| Sleeve 2        | 10    | $t_1$       | 3.8 | 2   |
| $t_{10}$| Gear 6             | 22    | $t_1$| 1.3  | 2   | $t_35$| Eccentric sleeve 2 | 8     | $t_{28}$    | 2.1 | 2   |
| $t_{11}$| Bear 8             | 36    | $t_1$| 2.1  | 2   | $t_36$| Lubricating ring 1 | 4     | $t_1$       | 1.5 | 2   |
| $t_{12}$| Lubricating ring 4  | 27    | $t_1$| 3.8  | 2   | $t_37$| Gear 1          | 7     | $t_1$       | 3.1 | 2   |
| $t_{13}$| Sleeve 3           | 29    | $t_{39}$, $t_4$ | 1.7 | 2   | $t_{38}$| Sleeve 1        | 6     | $t_1$       | 3.8 | 2   |
| $t_{14}$| Inner splined sleeve | 19    | $t_3$| 2.8  | 2   | $t_{39}$| Eccentric sleeve 1 | 46    | $t_{29}$    | 2.3 | 2   |
| $t_{15}$| End cap 4          | 13    | $t_6$| 2.1  | 2   | $t_{40}$| Lock 2          | 34    | $t_{30}$    | 1.9 | 2   |
| $t_{16}$| Gear 5             | 14    | $t_{7}$, $t_6$ | 3.1 | 2   | $t_{41}$| Middle box      | 11    | $t_{31}$, $t_{24}$, $t_39$ | 2.1 | 2   |
| $t_{17}$| Gear 4             | 25    | $t_1$| 3.6  | 2   | $t_{42}$| Lock 1          | 32    | $t_{25}$    | 1.6 | 2   |
| $t_{18}$| Gear 3             | 21    | $t_{6}$, $t_{10}$ | 3.5 | 2   | $t_{43}$| Input shaft      | 43    | $t_{31}$, $t_32$ | 4.2 | 2   |
| $t_{19}$| Bear 7             | 38    | $t_{11}$| 2.1 | 2   | $t_{44}$| Output shaft 1   | 5     | $t_{33}$    | 7.2 | 2   |
| $t_{20}$| Blank cap 2        | 26    | $t_{12}$| 0.9 | 2   | $t_{45}$| End cap 3       | 37    | $t_{33}$    | 0.9 | 1   |
| $t_{21}$| Gear 2             | 30    | $t_{13}$| 3.8 | 2   | $t_{46}$| End cap 3       | 37    | $t_{39}$, $t_{32}$, $t_{41}$ | 0.9 | 1   |
| $t_{22}$| Bear 6             | 31    | $t_1$| 1.9  | 2   | $t_{47}$| End cap 2       | 12    | $t_1$       | 0.9 | 1   |
| $t_{23}$| Bear 5             | 40    | $t_1$| 2.1  | 2   | $t_{48}$| End cap 1       | 35    | $t_{40}$    | 1.8 | 1   |
| $t_{24}$| Intermediate shaft | 16    | $t_{14}$, $t_{19}$ | 9.5 | 2   | $t_{49}$| Upper box       | 49    | $t_{42}$, $t_{43}$ | 4.1 | 1   |
| $t_{25}$| Output shaft 2     | 28    | $t_{20}$, $t_{23}$ | 12.1 | 2   | $t_{50}$| Oil collecting hood | 1     | $t_{44}$ | 1.6 | 1   |

**Figure 4: SAPN model of the reducer.**
Step 6: If the termination condition is satisfied, decode the optimal solution and end. Otherwise, return to step 2.

In the optimization procedure, the parameters are set as $\delta_p = 5$, $\delta_l = 5$, $c_l = c_g = 2$, $d_l = 0.1$, and $d_l = 0.001$, and $\omega$ will linearly decrease from 1 to 0 during the searching process. Here, the parameters are set not only based on the features of PASP problem and PSOBC algorithm, but also referring to researches in [25, 26].

4.4. Discussions and Comparisons. Three different simulations are conducted to verify the proposed method when DPAs are settled as 2, 3, and 4, respectively. The proposed PSOBC simulations have been compared with standard PSO method in Figure 7. To keep consistency, the simulation parameters including learning factor, penalty factors, and inertia factor are chosen as the same in both PSOBC and PSO for different DPAs. In Figure 7(a), DPAs are both settled as 2. For PSO simulation, the assembly time is 81.5 seconds. The assembly time of PSOBC method is 74.8 seconds. Compared to the standard PSO, the efficiency is increased by 9.0%. Detailed explanation of 2-DPA assembly sequence is given in Table 2.

In Figure 7(b), DPAs are both settled as 3. It is seen that the assembly time of PSO is 62.1 seconds while the assembly time of PSOBC is 59.6 seconds. Compared to the standard PSO, the assembly efficiency is increased by 4.2%. The
optimal sequence carried out by PSOBC method is given in Table 3 when DPA is set as 3.

In Figure 7(c), DPAs are both settled as 4. Through standard PSO method, assembly time of the solution is 59.1 seconds. The assembly efficiency is increased by 3.1%. The optimized sequence carried out by PSOBC is given in Table 4.

Based on the optimization results and comparison, the following conclusions can be drawn:

(1) Due to implementation of SAPN model, the PASP problem can be effectively and efficiently solved for complex products, while huge matrix calculation is avoided and possibility of human error is reduced.

(2) Compared with the standard PSO algorithm, premature convergence can be effectively prevented through repulsion operation. The solution of PASP problem can be further optimized.

![Figure 7: Optimization results and comparison. (a) Optimization result with 2-DPD. (b) Optimization result with 3-DPD. (c) Optimization result with 4-DPD.](image)

| Subassembly index | Subassembly | Targeted parts |
|-------------------|-------------|----------------|
| 1                 | (17, 20)    | Bear 9 and spline |
| 2                 | (23, 15)    | Gear 7 and bear 10 |
| 3                 | (22, 36)    | Gear 6 and bear 8 |
| 4                 | (21, 14)    | Gear 13 and gear 5 |
| 5                 | (40, 18)    | Gear 7 and bear 12 |
| 6                 | (45, 24)    | Bear 11 and eccentric sleeve 4 |
| 7                 | (38, 13)    | Bear 7 and end cap 4 |
| 8                 | (19, 27)    | Inner splined sleeve |
| 9                 | (41, 44)    | Lubricating ring 3 and eccentric sleeve 5 |
| 10                | (26, 29)    | Blank cap 2 |
| 11                | (31)        | Bear 6 |
| 12                | (30, 25)    | Gear 2 and gear 4 |
| 13                | (28, 16)    | Output shaft and intermediate shaft |
| 14                | (32, 3)     | Lock 1 and bear 4 |
| 15                | (42, 47)    | Lubricating ring 2 and bear 2 |
| 16                | (39)        | Bear 1 |
| 17                | (46, 6)     | Eccentric sleeve 1 and sleeve 1 |
| 18                | (33, 9)     | Lock 3 and bear 3 |
| 19                | (34, 8)     | Lock 2 and eccentric sleeve 2 |
| 20                | (7)         | Gear 1 |
| 21                | (43, 10)    | Input shaft and sleeve 2 |
| 22                | (2)         | Sleeve 3 |
| 23                | (11, 4)     | Middle box and lubricating ring 1 |
| 24                | (12, 37)    | End cap 2 and end cap 3 |
| 25                | (5, 49)     | Output shaft 1 and upper box |
| 26                | (1, 35)     | Oil collecting hood |
| 27                | (48)        | Blank cap 1 |

(1) Due to implementation of SAPN model, the PASP problem can be effectively and efficiently solved for complex products, while huge matrix calculation is avoided and possibility of human error is reduced.

(2) Compared with the standard PSO algorithm, premature convergence can be effectively prevented through repulsion operation. The solution of PASP problem can be further optimized.
5. Conclusions

A novel PASP optimization method is developed for complex product in this paper. A SAPN model is proposed to describe the precedence relationships in the assembly process, while huge matrix calculation is avoided and possibility of human error is reduced. To optimize PASP problem, a PSOBC method is developed along with the SAPN model. Verifications show that assembly time of the optima calculated using the proposed method is 9.0%, 4.2%, and 3.1% better than the standard PSO algorithm when the DPA is 2, 3, and 4, respectively. Assembly time of the three optimal sequences is 74.8 seconds, 59.6 seconds, and 57.3 seconds, separately. Parallel assembly process can be optimally planned by using the proposed method.

Nomenclature

Sets
- \( P \): Finite set of places
- \( T \): Finite set of transitions
- \( F \): Finite set of arcs
- \( W \): Finite set of weights
- \( M \): Finite set of markings
- \( pt \): Finite set of preceding transitions
- \( PT \): Finite set of \( pt \)s
- \( TM \): Finite set of operation times

Indices
- \( i, i_1, i_2 \): Indices of transitions
- \( j \): Index of places
- \( k \): Index of arcs
- \( g \): Index of iterations
- \( s \): Index of subassemblies
- \( d, d_1, d_2 \): Indices of particles

Symbols
- \( p \): Place
- \( t \): Transition
- \( f \): Arc
- \( w \): Weight
- \( m \): Marking
- \( tm \): Operation time of each transition

Table 3: Parallel assembly sequence of 3-DPA.

| Subassembly index | Subassembly | Targeted parts                                      |
|-------------------|-------------|-----------------------------------------------------|
| 1                 | (44, 45, 18)| Eccentric sleeve 5, bear 11, and bear 12           |
| 2                 | (29, 27, 23)| Sleeve 3, lubricating ring 4, and gear 7           |
| 3                 | (26, 21, 22)| Blank cap 2, gear 3, and gear 6                    |
| 4                 | (9, 3, 31)  | Bear 3, gear 4, and bearing                        |
| 5                 | (6, 24, 20) | Sleeve 1, eccentric sleeve 4, and spline           |
| 6                 | (15, 17, 19)| Bear 10, bear 9, and inner splined sleeve          |
| 7                 | (7, 2, 36)  | Gear 1, sleeve 3, and bear 8                       |
| 8                 | (30, 38, 25)| Gear 2, bear 7, and gear 4                         |
| 9                 | (4, 8)      | Lubricating ring 1 and eccentric sleeve 2          |
| 10                | (14, 10, 13)| Gear 5, sleeve 2, and end cap 4                   |
| 11                | (39, 40)    | Bear 1 and bear 5                                  |
| 12                | (5, 28, 16) | Output shaft 1                                    |
| 13                | (32, 42, 46)| Lock 1, lubricating ring 2, and eccentric sleeve 1|
| 14                | (43, 11, 34)| Input shaft, middle box, and lock 2               |
| 15                | (49, 1, 37) | Upper box, oil collecting hood, and end cap 3      |
| 16                | (35, 12, 48)| End cap 1, end cap 2, and blank cap 1             |

Table 4: Parallel assembly sequence of 4-DPA.

| Subassembly index | Subassembly | Targeted parts                                      |
|-------------------|-------------|-----------------------------------------------------|
| 1                 | (27, 17)    | Lubricating ring 4 and bear 9                       |
| 2                 | (26, 15, 20, 18) | Blank cap 2, bear 10, spline, and bear 12  |
| 3                 | (19, 36, 24) | Inner splined sleeve, bear 8, and eccentric sleeve 4|
| 4                 | (38, 45, 44, 31)| Bear 7, bear 11, eccentric sleeve 5, and bear 6   |
| 5                 | (23, 29, 4, 22)| Gear 7, sleeve 3, lubricating ring 1, and gear 6   |
| 6                 | (30, 21, 14) | Gear 2, gear 3, and gear 5                          |
| 7                 | (25, 13)    | Gear 4 and end cap 4                                |
| 8                 | (28, 16)    | Output shaft 2 and intermediate shaft               |
| 9                 | (7, 3, 47, 40)| Gear 1, bear 4, bear 2, and bear 5                 |
| 10                | (46, 2, 39, 41)| Eccentric sleeve 1, sleeve 3, bear 1, and lubricating ring 3|
| 11                | (9, 42, 32, 11)| Bear 3, lubricating ring 2, lock 1, and middle box|
| 12                | (8, 33, 43) | Eccentric sleeve 2, lock 3, and input shaft        |
| 13                | (34, 6, 10) | Lock 2, sleeve 1, and sleeve 2                     |
| 14                | (49, 37, 5) | Upper box, end cap 3, and output shaft 1            |
| 15                | (35, 1, 48, 12)| End cap 1, oil collecting hood, blank cap 1, and end cap 2|
Acknowledgments

This work was supported in part by the Zhejiang Key Research and Development Program under Grant 2017C01008, the Analytical and Testing Technology Project of Zhejiang Province under Grant 2018C37068, the Key Technology Research Project of Wenzhou under Grants 2018ZG020 and ZG2017002, and the Open Research Fund of the Low-Voltage Apparatus Technology Research Center of Zhejiang under Grant 201711-01.

References

[1] B. Deepak, M. R. G. Bala Murali, and B. Biswal, “Assembly sequence planning using soft computing methods: a review,” Proceedings of the Institution of Mechanical Engineers, Part E: Journal of Process Mechanical Engineering, vol. 233, no. 3, pp. 653–683, 2019.
[2] M. R. Bahubalendruni and B. B. Biswal, “A review on assembly sequence generation and its automation,” Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science, vol. 230, no. 5, pp. 824–838, 2016.
[3] L. S. De Mello and A. C. Sanderson, “AND/OR graph representation of assembly plans,” IEEE Transactions on Robotics and Automation, vol. 6, no. 2, pp. 188–199, 1990.
[4] Y. Yang, P. Yang, J. Li et al., “Research on virtual haptic disassembly platform considering disassembly process,” Neurocomputing, vol. 348, pp. 74–81, 2019.
[5] Y.-j. Wu, Y. Cao, and Q.-f. Wang, “Assembly sequence planning method based on particle swarm algorithm,” Cluster Computing, vol. 22, no. S1, pp. 835–846, 2019.
[6] Y. Wang and D. Tian, “A weighted assembly precedence graph for assembly sequence planning,” The International Journal of Advanced Manufacturing Technology, vol. 83, no. 1–4, pp. 99–115, 2016.
[7] A. Bedeoui, R. B. Hadji, M. Hammadi et al., “Assembly sequence plan generation of heavy machines based on the stability criterion,” The International Journal of Advanced Manufacturing Technology, 2019.
[8] W. Pan, Y. Wang, and X.-D. Chen, “Domain knowledge based non-linear assembly sequence planning for furniture products,” Journal of Manufacturing Systems, vol. 49, pp. 226–244, 2018.
[9] M. Bahubalendruni, A. Gulivindala, S. Varupala et al., “Optimal Assembly Sequence Generation through computational approach,” Sādhanā, vol. 44, no. 8, p. 174, 2019.
[10] M. V. A. R. Bahubalendruni, A. Gulivindala, M. Kumar, B. B. Biswal, and L. N. Annapu, “A hybrid conjugated method for assembly sequence generation and explode view generation,” Assembly Automation, vol. 39, no. 1, pp. 211–225, 2019.
[11] M. R. Bahubalendruni and B. B. Biswal, “An intelligent approach towards optimal assembly sequence generation,” Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science, vol. 232, no. 4, pp. 531–541, 2018.
[12] L. Zhang, H. Lv, D. Tan et al., “Adaptive quantum genetic algorithm for task sequence planning of complex assembly systems,” Electronics Letters, vol. 54, no. 14, pp. 870–872, 2018.
[13] H. S. Wang, C. H. Tu, and K. H. Chen, “Supplier selection and production planning by using guided genetic algorithm and dynamic nondominated sorting genetic algorithm II approaches,” Mathematical Problems in Engineering, vol. 2015, pp. 1–15, 2015.
[14] M. Zhang, L. Wang, Z. Cui et al., “Fast nondominated sorting genetic algorithm II with levy distribution for network topology optimization,” Mathematical Problems in Engineering, vol. 2020, Article ID 3094941, 12 pages, 2020.
[15] X. Hu, Z. Xu, L. Yang et al., “A novel assembly Line Scheduling Algorithm based on CE-PSO,” Mathematical Problems in Engineering, vol. 2015, Article ID 685824, 9 pages, 2015.
[16] R. Ab, F. Mohd, A. Tiwari et al., “Integrated optimization of mixed-model assembly sequence planning and line balancing using Multi-Objective Discrete Particle Swarm

Variables

time: Operation time of assembly/subassembly
X: Vector of particle position
V: Vector of particle velocity
FT (X): Fitness function
Fp (s): Penalty function for violating precedence relationships
Fp (s): Penalty function for violating parallel principles
Fd (P): Diversity function

Abbreviations

SAPN: Synchronized assembly Petri net
ASP: Assembly sequence planning
PASP: Parallel assembly sequence planning
SASP: Sequential assembly sequence planning
PSO: Particle swarm optimization
PSOBC: Particle swarm optimization based on bacterial chemotaxis.

Data Availability

No data were used to support this study.

Conflicts of Interest

The authors declare that they have no conflicts of interest.
Optimization,” *Artificial Intelligence for Engineering Design, Analysis and Manufacturing: AIEDAM*, vol. 33, no. 3, pp. 332–345, 2019.

[17] Y. Zhao, W. Li, X. Wang et al., “Path planning of slab library crane based on improved ant colony algorithm,” *Mathematical Problems in Engineering*, vol. 2019, Article ID 7621464, 16 pages, 2019.

[18] J. Huo, Z. Wang, F. Chan et al., “Assembly line balancing based on beam ant colony optimisation,” *Mathematical Problems in Engineering*, vol. 2018, Article ID 2481435, 17 pages, 2018.

[19] J.-F. Tsai, J. G. Carlsson, D. Ge, Y.-C. Hu, and J. Shi, “Improved quantum-inspired evolutionary algorithm for engineering design optimization,” *Mathematical Problems in Engineering*, vol. 2012, pp. 1–7, 2012.

[20] W. Hongbin, D. Jian, and W. Yueling, “High-order feedback iterative learning control algorithm with forgetting factor,” *Mathematical Problems in Engineering*, vol. 2015, Article ID 826409, 7 pages, 2015.

[21] M. F. F. Ab Rashid, W. Hutabarat, and A. Tiwari, “Multi-objective discrete particle swarm optimisation algorithm for integrated assembly sequence planning and assembly line balancing,” *Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture*, vol. 232, no. 8, pp. 1444–1459, 2018.

[22] L. Shu, M. Dapino, G. Wu, and D. Chen, “Frequency-dependent sliding-mode control of Galfenol-driven unimorph actuator based on finite-element model,” *IEEE Transactions on Industrial Electronics*, vol. 63, no. 2, pp. 1071–1082, 2016.

[23] Y. Wang and J. H. Liu, “Chaotic particle swarm optimization for assembly sequence planning,” *Robotics and Computer-Integrated Manufacturing*, vol. 26, no. 2, pp. 212–222, 2010.

[24] B. Niu, Y. L. Zhu, X. X. He et al., “An improved particle swarm optimization based on bacterial chemotaxis,” *Proceedings of the World Congress on Intelligent Control and Automation (WCICA)*, vol. 1, pp. 3193–3197, 2006.

[25] Y. Cheng and Y. Ren, “Heuristic search solvers for differential game model of CO₂ emission in electricity market,” *Journal of Computational Information Systems*, vol. 8, no. 19, pp. 8151–8158, 2012.

[26] X. Y. Zhang, C. Li, and Z. Li, “Optimal reactive power dispatch based on mixed bacterial chemotaxis algorithm,” *Applied Mechanics and Materials*, vol. 494-495, pp. 1849–1852, 2014.