A Hybrid Estimation of Distribution Algorithm for Distributed Permutation Flowshop Scheduling with Flowline Eligibility

Wenzhe Duan, Zhengyang Li, Mengchen Ji, Yixin Yang, Shouyang Wang, Bo Liu
Academy of Mathematics and Systems Science
Chinese Academy of Sciences
Beijing 100190, China
duanwenzhe14@mails.ucas.ac.cn (WZD), liub01@mails.tsinghua.edu.cn (BL)

***Abstract*—This paper studies a new extension of distributed permutation flow shop scheduling problem (DPFSP) referred to as the DPFSP with flowline eligibility. Under this extension, the problem contains two stages. At the first assignment stage, a set *N* of *n* jobs is distributed among *F* factories with flowline eligibility constraint, i.e., not all the factories are available for every job. Then each factory can be regarded as a regular permutation flow shop scheduling problem (PFSP) in which jobs assigned to each factory have to be processed on *m* machines according to the given permutation. The objective is to minimize the maximum completion time or makespan of all the factories. The flowline eligibility constraint adds asymmetry and complexity to the regular DPFSP. In the proposed hybrid estimation of distribution algorithm (hybrid EDA), a heuristic of Framinan and Leisten (FLH) based solution-generate method (FLHSGM), two novel factory-based operators are proposed, and an effective local search based on variable neighborhood search (VNS) is employed and incorporated into EDA. Numerical simulations with comprehensive computational and statistical analysis are carried out. The experimental results and comparisons with existing algorithms show that the FLHSGM and the VNS-based local search enhances the search ability of EDA and the effectiveness of the hybrid EDA in solving both small-scale and large-scale DPFSP with flowline eligibility.**

Keywords*—*DPFSP with flowline eligibility constraint, EDA, VNS

I. INTRODUCTION

Production scheduling plays an important role in the decision making of a manufacturing system. Significant research efforts in the development of optimization approaches have been made, and some optimization frameworks and algorithms have been proposed [1-3]. The distributed permutation flow-shop scheduling (DPFSP) has been a very active and prolific combinatorial problem proposed by Naderi and Ruiz [4]. The regular DPFSP is a two-stage problem. In the first assignment stage, a set of *n* jobs are distributed among *F* factories each of which contains *m* machines. All factories are available for processing all jobs. Then each factory can be regarded as a permutation flow-shop scheduling problem (PFSP) where jobs assigned to each factory have to be processed on *m* machines according to the given permutation. Many extensions of regular DPFSP have be proposed. For instances, Hatami et al. proposed the distributed assembly permutation flow shop scheduling problem (DAPFSP) in 2013 with the background of complex supply chains [5]. This extension added an assembly stage on DPFSP where all the jobs have to be assembled into final products after being produced in the factory. An extended DPFSP model was designed by Li et al in 2016, which considers transportation times and loading capacities of factories [6].

As mentioned above, there is a common and universal assumption of the regular DPFSP and of all the extension model of DPFSP: all the factories, production centers or shops are available for all of jobs [4]. This assumption means that each job can be processed in any factory with no restriction. However, this situation is less common. Due to technological or transportation constraints, machines cannot be utilized for some jobs and the cost for transporting some jobs to some factories may be unacceptable. Thus some jobs must be processed only in certain factories which is described as eligibility constraint. The eligibility constraint on parallel machine scheduling has already been studied by Hwang and Chang [7], and Kellerer [8]. In 2015, Hatami et al. proposed DAPMSP with eligibility constraint [9]. In this paper, we studies a new extension of distributed permutation flow-shop scheduling problem referred to as DPFSP with flowline eligibility. This extension adds a constraint to the first assignment stage. For each job *i* there is a set M_i which denotes the unavailable factories of job *i*. All jobs must be assigned to the factory they are allowed to. The second processing stage of DPFSP with flowline eligibility is the same as the regular DPFSP.

The regular DPFSP is a kind of NP-hard problem [9]. After the pioneer work of Naderi and Ruiz, many intelligent optimization algorithms have been utilized for the DPFSP and its extensions. For example, a modified iterated greedy algorithm [10] and a scatter search algorithm [11] was proposed for regular DPFSP. Wang and Wang proposed an EDA-based memetic algorithm which performed well in

This research is partially supported by National Natural Science Foundation of China (Grant No. 71101139 and 71390331), as well as Defense Industrial Technology Development Program.

978-1-5090-0623-6/16/$31.00 ©2016 IEEE

DAPFSP [12]. Because of the flowline eligibility constraint, the asymmetric nature of the DPFSP with flowline eligibility is stronger than the regular DPFSP. This asymmetry adds much complication upon problem solving. In this paper, a hybrid estimation of distribution algorithm (EDA) is proposed for DPFSP with flowline eligibility constraint. As a population-based evolutionary algorithm, EDA draws much attention in the field of intelligent optimization [13]. Due to the ability of solving high-dimensional optimization problems, EDA has become an essential asset in the algorithmic toolbox [14]. The main characteristic of EDA is its probability model, which is updated with the evolution of the population and adds the search probability of promising solutions. Besides, to execute the searching operator efficiently, decrease the scale of the searching space and adapt to the constraint of flowline eligibility, a novel factory-based method is proposed to represent a solution for the DPFSP with flowline eligibility constraint. This method uses only one layer encoding which is denoted as a vector *s*. The *i*th element of *s* is the assigned factory of *i*th job. Then the permutation of jobs in each factory is generated by a heuristic first proposed by Framinan and Leisten [15]. Since variable neighborhood search (VNS) is the most preferred algorithm in the literature with which hybridize [16] , in the hybrid EDA, an effective local search based on VNS [17] is put forward to balance the exploration and exploitation. And two novel factory-based operators are employed and incorporated into VNS in order to avoid invalid searching operators.

The remainder of this paper is organized as follows: In Section II, the DPFSP with flowline eligibility constraint is described; Then the hybrid EDA for DPFSP with flowline eligibility constraint is introduced in Section III; In Section IV, experimental results on benchmarks together with comparisons are presented and analyzed; Finally, the conclusion is given in Section V.

II. DPFSP with Flow Line Eligibility Constraint

The DPFSP with flowline eligibility constraint can be defined as a two-stage problem. a set *N* of *n* jobs, denote as $\left\{job_1, job_2, \cdots, job_n\right\}$, have to be processed on a set *G* of *F* factories, each one of them contains the same set *M* of *m* machines. The unavailable factory set of job_i is denotes as M_i:

- Factory *f* belongs to M_i, $f \in M_i$, means that the *f*th factory is not available for job_i;
- The set M_i is empty means that all factories is capable for *i*th job;
- M_i cannot be the set of all factories, i.e., $M_i \neq G$.

At the first assignment stage, all jobs are distributed among *F* factories and each job must be assigned to its available factors. Once job_i is assigned to factory *f*, it cannot be transferred to another factory as it must be completed in one factory during the second processing stage. The processing times of the job_i's operation on *j*th machine of all the factory are the same, i.e., $T_{ij}^1 = T_{ij}^2 = \cdots = T_{ij}^F = T_{ij}$. Besides, the following assumptions for the regular DPFSP scheduling are adopted: all the jobs are independent and available at time 0. Each machine can process only one job at a time and each job can be processed by only one machine at a time. Each operation must be completed without interruption once it is started. The DPFSP with flowline eligibility constraint is to assign jobs to factories under constraint and to determine the job permutation for all machines of each factory to optimize a certain scheduling objective. In this paper, the objective is to minimize the maximum completion time of each factories.

Denote $h^f = \left(job^f(1), job^f(2), \cdots, job^f(n_f)\right)$ as the job permutation in factory *f* in the processing stage where n_f is the total number of jobs which is assigned to factory *f*. For any $job^f(i) \in h^?$, the factory *f* must be available for it, i.e., $f \notin M_{job^f(i)}$. A feasible solution *h* of DPFSP with flowline eligibility constraint can be represented as a list of n_f job permutation vectors. Denotes C_{ij} as the completion time that job_i finishes the operator on *j*th machine. For a feasible solution *h*, the total completion time or makespan is calculated as follows:

$$C_{job^f(1),1} = T_{job^f(1),1} \tag{1}$$

$$C_{job^f(i),1} = C_{job^f(i-1),1} + T_{job^f(i),1} \tag{2}$$

$$C_{job^f(1),j} = C_{job^f(1),j-1} + T_{job^f(1),j} \tag{3}$$

$$C_{job^f(i),j} = max\left(C_{job^f(i),j-1}, C_{job^f(i-1),j}\right) + T_{job^f(i),j} \tag{4}$$

where $f = 1, 2, \ldots, F$, $i = 1, 2, \ldots, n_f$, $j = 1, 2, \ldots, m$.

The objective of solving the DPFSP with flowline eligibility constraint is to find a feasible solution *h* with the minimum makespan. However, as an NP-hard problem, this problem cannot be solved by any polynomial time algorithms. Therefore, the objective of the solution methods is to obtain an approximate optimal feasible solution within an acceptable computing time.

III. HYbrid EDA for DPFSP with Flowline Eligibility Constraint

A. Brief Review of EDA

Estimation of distribution algorithm (EDA) is a population-based evolutionary optimization algorithm which explores the space of solutions by sampling an explicit probabilistic model constructed from promising solutions found so far [13]. It has already been successfully applied to solve different kinds of academic and practical problems. In scheduling field, the EDA has been utilized to solve the flow-shop problem [19], hybrid flow-shop scheduling problem [20], flexible job-shop scheduling problem [21] and so on. The EDA based MA has also solved the regular DPFSP successfully [12].

EDA works with a population of candidate solutions to the problem, starting with the population generated according to the uniform distribution over all admissible solutions. The population is then scored using a fitness function. In DPFSP with flowline eligibility constraint, this fitness function is the maximum completion time of a feasible solution. Then the population is sorted by their scores. From this ranked population, a subset of the most promising solutions are selected by the selection operator. A simple but useful selection operator is truncation selection with threshold τ = 50%, which selects the 50% best solutions among all the feasible solutions in one generation. The probability model then can be updated according to the information obtained from these superior solutions. Once the model is constructed, new solutions of the next generation are generated by sampling the distribution encoded by this model. These new solutions are then incorporated back into the old population, possibly replacing it entirely. This process is repeated until a solution of sufficient quality is reached or the number of iterations reaches some threshold, with each iteration of this procedure usually referred to as one generation of the EDA.

B. *Solution Representation*

Solution representation is a key issue for every metaheuristic method. In the regular DPFSP, a set of *F* lists or the two layer encoding based method is the most widely used representation. In the DPFSP with flowline eligibility constraint, we represent every feasible solution as a set *h* as same as DPFSP and a factory assignment vector *s*, each list of *h* contain a job permutation vector with the order in which jobs will be processed at each factory. This kind of representation is the most straightforward method, for instance, if there is a problem with 10 jobs ($n = 10$) and three factories ($F = 3$), and $M_1 = \left\{1, 3\right\}$, $M_2 = \left\{2\right\}$, the sets of unavailable factories of the other jobs are empty. One feasible solution is :

$$h = \begin{cases} 10, 4, 2, 8 \\ 1, 9, 6 \\ 5, 3, 7 \end{cases} \tag{5}$$

and

$$s = \left\{2, 1, 3, 1, 3, 2, 3, 1, 2, 1\right\} \tag{6}$$

In order to execute the searching operator efficiently and decrease the scale of searching space, in this paper we proposed a novel solution-generate method, which makes it possible to generate a new feasible solution *h* from a given factory assignment vector *s*. This novel method is based on a heuristic for total flow time minimization in permutation flow shops, which is first proposed by Framinan and Leisten (FLH) [15]. The details of FLH based solution-generate method will be described as follows:

Step 1: For each factory *f*, let

$$S^f = \left\{job_{i_1}^f, job_{i_2}^f, \dots, job_{i_n}^f\right\} \tag{7}$$

represents the set of jobs assigned to factory *f* and is calculated from the factory assignment vector *s*;

Step 2: For job job_i^f each set S^f , calculate the total processing time T_i^f which is given by:

$$T_i^f = \sum_{j=1}^{M} T_{ij} \tag{8}$$

then sort all jobs of S^f in ascending order of T_i^f ;

Step 3: Set $k = 2$, Select the first two jobs from the sorted list and choose the better between the two feasible sequences as the best partial sequence so far.

Step 4: let $k = k + 1$. Choose the *k*th job from the sorted list and insert it into the *k* possible positions of the temporary best partial sequence. The one with minimum completion time is selected as the best partial sequence obtained so far;

Step 5: If $k = n$, stop, let $f = f + 1$ and go to *Step* 1; else, go to *Step* 4.

C. *Probability Model*

The most important part of the EDA is its probability model that describes the distribution of the searching space. Generally, the construction of the probability model is a statistical learning process based on the information of superior feasible solutions. Besides, the EDA generates new solutions by sampling the distribution which is obtained from the probability model. Therefore, a proper and explicit probability model is critical to the performance of EDA-based algorithms.

In this paper, the optimization objective is to minimize the complement time of the DPFSP with flowline eligibility. Both factory assignment vector and permutation of jobs in each factory play an important role in completion time. However, according to above solution representation, the permutation of jobs in each factory can be obtained by the factory assignment vector. To decrease the scale of probability model, the probability model is designed as a probability matrix, which is related to the factory assignment vector of a feasible solution:

$$p(g) = \begin{pmatrix} p_{11}(g) & p_{12}(g) & \cdots & p_{1F}(g) \\ p_{21}(g) & p_{22}(g) & \cdots & p_{2F}(g) \\ \vdots & \vdots & & \vdots \\ p_{n1}(g) & p_{n2}(g) & \cdots & p_{nF}(g) \end{pmatrix} \tag{9}$$

Element $p_{ij}(g)$ of the probability matrix *P* represents the probability that the *i*th job is assignment to the *j*th factory during the *g*th generation, i.e., *i*th element of a feasible factory assignment vector appears. At the initialization stage of the hybrid EDA, the elements in matrix P are initialized as:

$$p_{ij}(0) = \begin{cases} \dfrac{1}{N - \#M_i} & , j \in M_i \\ 0 & , j \notin M_i \end{cases} \tag{10}$$

for all i and j, where $\#M_i$ denotes the size of set M_i. It implies a uniform distribution in the searching space and no useful prior knowledge is available.

The probability model must be well adjusted to make the search procedure tract the promising searching region. An updating mechanism based on the information of the superior feasible solutions is employed to adjust the probability model during each generation. This updating mechanism can be regarded as a kind of learning process as follows:

$$p_{ij}(g+1) = (1-\alpha)p_{ij}(g) + \frac{\alpha}{S\_POP}\sum_{k=1}^{S\_POP} I_{ij}^{k}(g) \tag{11}$$

where, $\alpha \in (0,1)$ denotes the constant learning rate of the probability model, S_POP is the number of superior feasible solutions and $S\_POP = \tau \times POP$. $I_{ij}^{k}(g)$ is the indicator function of kth superior solutions in gth generation. $I_{ij}^{k}(g) = 1$ means that ith job of kth superior solutions is assigned to the jth factory; $I_{ij}^{k}(g) = 0$, otherwise.

D. EDA-based Search

In this paper, the distribution information of superior feasible solutions can be evolved by EDA-based searching operator, (9), (10) and (11) are employed to initialize and construct the probability model of EDA and to describe the distribution of the searching space. Note the, the DPFSP with flowline eligibility constraint is a complex problem with strong asymmetry. Aimed at executing the searching operator efficiently and decreasing the scale of searching space, a FLH-based solution-generate method is incorporated into EDA. In the following contents, we will present the VNS-based local search that is incorporated in EDA to propose a hybrid EDA.

E. VNS-based Local Search

Variable neighborhood search (VNS) is a metaheuristic, or a framework for building heuristics, based upon systematic changes of neighborhoods both in descent phase, to find a local minimum, and in perturbation phase to emerge from the corresponding valley. It was first proposed in 1997 by Mladenovic and Hansen and has since then rapidly developed both in its methods and its applications. VNS heavily relies on the following three facts [17]:

- A local minimum with respect to one neighborhood structure is not necessarily a local minimum for another neighborhood structure;
- A global minimum is a local minimum with respect to all possible neighborhood structures;
- For many problems local minima with respect to one or several neighborhoods are relatively close to each other.

A variety of scheduling problems has been efficiently solved with VNS approach. Lin and Ying proposed a hybrid Tabu-VNS approach for single machine tardiness problem with sequence-dependence setup times [22]. A VNS-based PSO was applied by Liu et al. to tackle the multi-objective flexible job-shop scheduling problems [23]. Zobolas et al. designed a hybrid GA-VNS approach to minimize the total completion time in PFSP [24].VNS method has been employed in the regular DPFSP and its extension problems and has performed well. For example, Naderi and Ruiz applied VNS for solving the DPFSP firstly [4]. VNS was used in Hatami et al. to find the optimal solutions in distributed assembly permutation flowshop scheduling problem (DAPFSP) [5].

In this paper, three job-based operators and two novel factory-based operators are employed and incorporated into VNS as different neighborhood of a feasible solution. The job-based operators are aimed at changing the job sequence inside each factory. Details of three job-based operators will show as follows:

1） Job-Swap: Swap the priorities of $job^{f}(i)$ and $job^{f}(j)$ in the job permutation vector of factory f;

2） Job-Invert: Invert the partial permutation between the positions of $job^{f}(i)$ and $job^{f}(j)$ in the job permutation vector of factory f;

3） Job-Insert: Partial permutation vector is obtained by removing one single $job^{f}(i)$ from the job sequence of factory f, insert $job^{f}(i)$ into all possible positions of the partial permutation vector and choose the feasible solution h with minimum completion time of factory f;

where, $job^{f}(i)$ and $job^{f}(j)$ denote the labels of jobs in factory f.

The two factory-based operators adjust both the job permutation vector in each factory and the factory assignment vector of a feasible solution. Most of the existing factory-based operators, such as the one designed by Naderi and Ruiz [4], change only one job's factory during an operation. It is possible that all the offspring solutions generated by these factory-based operators are worse than their parents. However, maybe one of the offspring solutions sits on the "key path" to the optimal solution, i.e., the optimal solution will be obtained from one of the offspring solutions during more than one operations. Unfortunately, all these offspring solutions are abandoned by VNS algorithm. In this paper, two novel factory-based operators are designed to avoid this. These two operators are able to change more than one jobs' factory in one operations. We pay much attention on the critical factory, which is the factory with maximum completion time. The critical path of DPFSP with flow line eligibility depends on the permutation of jobs in the critical factory. Thus an effective way to improve the solution of DPFSP with flow line eligibility is to remove or change one job in critical factory. Details of two factory-based operators will show as follows:

4） Factory-Cycle: Generate a job vector $\left(job^{f_1}(i_1), job^{f_2}(i_2), \dots, job^{f_{m_c}}(i_{m_c})\right)$, in which $f_1 \notin M_{job^{f_2}(i_2)}, f_2 \notin M_{job^{f_2}(i_3)}, \dots, f_{m_c} \notin M_{job^{f_1}(i_1)}$ and $f_1 \neq f_2 \neq \cdots \neq f_{m_c}, f_1 = f^c$. Then for each job permutation vector of factory f_k, remove $job^{f_k}(i_k)$, insert $job^{f_{k+1}}(i_{k+1})$ into all the possible positions and choose the job permutation vector with minimum completion time.

5） Factory-Sequence: Generate a factory label f_0 and a job vector $\left(job^{f_1}(i_1), job^{f_2}(i_2), \dots, job^{f_{m_s}}(i_{m_s})\right)$, in which $f_1 \notin M_{job^{f_2}(i_2)}, \dots, f_{m_{s-1}} \notin M_{job^{f_{m_s}}(i_{m_s})}$ and $f_0 \notin M_{job^{f_1}(i_1)}$, $f_0 \neq f_1 \neq \cdots \neq f_{m_s}$, $f_0 = f^c$. Then for each job permutation vector of factory $f_k, k = 1, 2, \dots, m_s$, remove $job^{f_k}(i_k)$, insert $job^{f_{k+1}}(i_{k+1})$ into all the possible positions and choose the job permutation vector with minimum completion time. Similarly, insert $job^{f_1}(i_1)$ into all the possible positions of the job permutation vector of factory f_0 and choose the one with minimum completion time.

where, $job^f(i)$ denotes the labels of jobs in factory *f*, f^c denotes the critical factory of the feasible solution. m^c and m^s denote as the length of cycle length and sequence length, respectively.

A solution h executes all the operators step by step. VNS based local search can be described as follows:

Step 1: Set $k = 1$ and $k_{max} = 5$;

Step 2: Execute the *k*th operator on solution *h*;

Step 3: If the new feasible solution h' obtained by Step 2 is accepted under some acceptance criteria, then the old solution is replaced and let $k = 1$; else let $k = k + 1$;

Step 4: If $k \leq k_{max}$, go to Step 2; else, stop.

The acceptance criteria is not the same between these two kinds of operators. We say a new feasible solution h' obtained by a job-based operator is acceptable if the *f*th factory's completion time of h' is shorter than *h*; on the other hand, a new feasible solution h' obtained by factory based operators is regarded as a better solution than *h* if total completion time of h' is shorter than the old solution *h*.

F. Hybrid EDA

In this sub-section, the Hybrid EDA is proposed for the DPFSP with flow line eligibility, where a novel solution-generate method based on FLH is applied to generate feasible solution from factory assignment vector, EDA and VNS-based local search are used for exploration and exploitation in search space respectively. The procedure of hybrid EDA is described as follows:

Step 1: Initialize the probability matrix *P*;

Step 2: Sample the probability matrix *P* and generate *POP* factory assignment vectors, denotes as *S*.

Step 3: For any $s \in S$, generate a feasible solution *h* with the FLH based solution-generate method;

Step 4: Evaluate all the feasible solutions, determine the best solution and the superior population;

Step 5: VNS-based local search on the best solution;

Step 6: Update the probability matrix *P* by the information of the superior population;

Step 7: If the best solution keep fixed at consecutive *L* steps, then output the best solution; else, go to Step 2.

IV. Numerical Results and Comparisons

To test the performance of the hybrid EDA, numerical tests are carried out by using two sets of benchmark instances of regular DPFSP with new flowline-eligibility constraint. All the benchmarks used in this paper are available at http://soa.iti.es. To evaluate the performance of the hybrid EDA, experimental results are evaluated by relative percentage deviation (RPD) as follows:

$$RPD = \frac{alg - opt}{opt} \times 100 \qquad (12)$$

where, *opt* is the solution of DPFSP with flowline eligibility constraint found by the comparing algorithms below and *alg* denotes to the makespan of the solution obtained by a certain algorithm. We also use BRE denote the relative error of the best with respect to the optimal solution of DPFSP with flowline eligibility constraint found so far.

A. Result and Comparsion of Small-scale Instances

In this section, by using 10 small-scale instances, we compare the hybrid EDA, the pure EDA and hybrid EDA2 in which the pure EDA is equipped with NEHSGM and VNS-based local search. The different between hybrid EDA and hybrid EDA2 is that the solution-generate method of hybrid EDA2 is NEHSGM instead of FLHSGM. In the solution-generate method based on NEH (NEHSGM), the job permutation vector of each factory is created by the well-known heuristic NEH instead of FLH. The following parameters are used: population size of EDA $POP = 20$; percentage of the superior sub-population τ = 50%; learning rate of probability matrix α = 0.1 and the stop parameter *L* = 40. Each instance run 10 times independently for every algorithms for comparison. The statistical performances of 10 independent benchmarks are listed in Table I, including the best relative error (BRE) and relative percentage deviation (RPD) to the optimal solution of DPFSP with flow line eligibility constraint found so far.

TABLE I. COMPARISON OF HYBRID EDA, PURE EDA AND HYBRID EDA2

Problem	Hybird EDA		Pure EDA		Hybrid EDA2	
	Average RPD	BRE	Average RPD	BRE	Average RPD	BRE
I_2_6_4_1	0	0	0	0	0	0
I_2_6_4_2	0	0	0	0	2.5000	2.5000
I_2_6_4_3	0	0	0	0	0	0
I_2_6_4_4	0	0	0	0	0	0
I_2_6_4_5	0	0	0	0	0	0
I_3_12_4_1	0	0	0.8403	0.5602	1.8768	1.1204
I_3_12_4_2	0	0	0	0	1.1538	0
I_3_12_4_3	0	0	0.2254	0	3.1831	0
I_3_12_4_4	0	0	0.6522	0.4831	2.5121	1.6909
I_3_12_4_5	0	0	0.4942	0.2907	2.6744	1.1628

From Table II, it can be seen that the proposed hybrid EDA provides the better optimization performance than other algorithms for all small-scale benchmarks. The optimal results obtained by the proposed hybrid EDA are all the same to the optimal solution of DPFSP with flowline eligibility found so far (that is to say, the BRE values are 0), which demonstrates the effective searching quality of the proposed hybrid EDA. On the other hand, the Average RPD values resulted by hybrid EDA are also zero, which demonstrates the robustness of the proposed algorithm in dealing with small-scale searching space. Simulation results and comparisons between hybrid EDA, pure EDA and hybrid EDA2 demonstrate that the FLH based solution-generate method (FLHSGM) and VNS-based local search are able to enhance the searching ability and balance between global search and local search of EDA.

B. *Result and Comparsion of Large-scale Instances*

Next, hybrid EDA is tested by using large-scaled instances. The following parameters are used: population size of EDA $POP = 20$; percentage of the superior sub-population $\tau = 50\%$; learning rate of probability matrix $\alpha = 0.1$ and the stop parameter $L = 40$. Considering number of factories, products, and jobs, a summarized result of average RPD and BRE is shown in Table II.

TABLE II. COMPARISON OF HYBRID EDA, PURE EDA AND HYBRID EDA2

Problem	Hybrid EDA		Pure EDA		Hybrid EDA2	
	Average RPD	BRE	Average RPD	BRE	Average RPD	BRE
Ta046_4	0.7938	0	3.0722	2.2628	2.3499	0.7833
Ta047_4	0.9814	0	3.4856	2.5381	2.7411	1.1844
Ta048_4	0.7340	0	3.6528	3.0225	2.6339	0.6045
Ta049_4	0.7755	0	3.6429	2.9757	2.5699	0.8115
Ta050_4	0.7432	0	3.3108	1.6892	3.2770	1.6047
Ta061_5	1.0561	0	3.5256	2.5995	2.0796	1.3810
Ta062_5	1.0802	0	3.4937	2.1097	2.4810	1.0970
Ta063_5	0.7106	0	3.1678	2.1404	1.4384	0.6849
Ta064_5	0.5212	0	3.1184	1.0601	1.4399	0.5300
Ta065_5	1.2119	0	4.0508	3.3051	3.0763	1.1864

Form Table III, it can be seen that hybrid EDA outperforms the other algorithms in solving all the groups of large-scaled instances. All the BRE values of Hybrid EDA are all 0 and the average RPD values are also very small, which demonstrates the effectiveness and robustness of the proposed hybrid EDA on large-scale searching space. In order to show the effectiveness of FLHSGM, we carry out simulation to compare our proposed hybrid EDA with the hybrid EDA2 which is defined above. In our simulation, the parameters of the two algorithms are the same and simulation results show that the proposed hybrid EDA outperforms hybrid EDA2 both in average RPD and BRE. On the other hand, the comparison between the proposed hybrid EDA and pure EDA demonstrates that the VNS-based local search are able to enhance the searching ability and balance between global search and local search of EDA.

C. *Wilcoxon Tests*

In this section, the pair-wise Wilcoxon test is used to compare the performance obtained by the proposed Hybrid EDA and the other two algorithms.

Table III shows the result of the comparisons of the our approach with both the Pure EDA and Hybrid EDA2 using 10 small-scale instances. Table IV shows the same comparison using 10 large-scale instances.

TABLE III. WILCOXON TEST OF SMALL-SCALE INSTANCES

Comparision	P_value (Ave.RPD)	P_value (BRE)
Hybrid EDA – Pure EDA	0.0350	0.0779
Hybrid EDA – Hybrid EDA2	0.0060	0.0350

TABLE IV. WILCOXON TEST OF LARGE-SCALE INSTANCES

Comparision	P_value (Ave.RPD)	P_value (BRE)
Hybrid EDA – Pure EDA	1.8267×10^{-4}	6.3864×10^{-5}
Hybrid EDA – Hybrid EDA2	1.8267×10^{-4}	6.3864×10^{-5}

In both cases we can observe that there is a significant difference between the proposed Hybrid EDA and the other two algorithms.

V. CONCLSION

In this paper we have studied an extension of the conventional distributed permutation flow shop scheduling problem (DPFSP) referred to as the DPFSP with flowline eligibility constraint. More specifically, we have proposed a new eligibility constraint on the first assignment stage of DPFSP. Under the flowline eligibility constraint, not all the factories are available for every jobs. Each job must be assigned to its feasible factory. Then each factory can be regarded as a regular permutation flow shop scheduling problem (PFSP) where jobs assigned to each factory have to be processed on *m* machines according to the given permutation. The objective is to minimize the maximum completion time, i.e., makespan of all the factories.

An effective hybrid EDA has been proposed for the DPFSP with flowline eligibility constraint. In the proposed algorithm, the EDA-based exploration and local search-based exploitation were well integrated. For the exploration phase, a probability model has been built to describe the probability distribution of superior feasible solutions. Besides, a heuristic of Framinan and Leisten based solution-generate method has been built in order to execute the searching operator efficiently, decrease the scale of the searching space and adapt to the constraint of flowline eligibility constraint. The effectiveness of the proposed FLHSGM has been shown by comparing with the NEH based solution generation and random order based solution generation. For the exploitation phase, two novel factory-based operators have been proposed and incorporated into the VNS based local search. Simulation results and comparisons with other algorithms have demonstrated that the VNS-based local search are able to enhance the searching ability of EDA both in small and large scale searching space. The results and comparison with existing algorithms have shown the effectiveness and efficiency of the proposed hybrid EDA.

ACKNOWLEDGMENT

The authors would like to thank the anonymous referees for their constructive comments on the earlier version of this paper. The last author BL is very grateful to Prof. Yew-Soon Ong (Nanyang Technological University, Singapore), Prof. Yihui Jin, Prof. Ling Wang (Tsinghua University), Prof. Jikun Huang (Peking University) and Royal Honored Prof. M.A. Keyzer (SOW-VU, Vrije Universiteit Amsterdam, The Netherlands) for their support.

REFERENCES

[1] B. Liu, L. Wang, and Y. H. Jin, "An effective PSO-based memetic algorithm for flow shop Scheduling," IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics, vol.37, no.1, pp.18–27, Feberrary 2007.

[2] B. Liu, L. Wang, Y. Liu, and S. Y. Wang, "A unified framework for population-based metaheuristics," Annals of Operations Research, Vol. 186, Iss. 1, pp 231–262, June 2011.

[3] B. Liu, L. Wang, Y. Liu, B Qian, and Y. H. Jin, "An effective hybrid particle swarm optimization for batch scheduling of polypropylene processes," Computers & Chemical engineering, Vol. 34, Iss. 4, pp. 518–528, April 2010.

[4] B. Naderi, and R. Ruiz, "The distributed permutation flowshop scheduling problem," Computers & Operations Research, Vol. 37, Iss. 4, pp. 754–768, April 2010.

[5] S. Hatami, R. Ruiz, and C. Andrés-Romano. "The distributed assembly permutation flowshop scheduling problem," International Journal of Production Research, Vol. 51, Iss. 17, pp. 5292–5308, September 2013.

[6] Z. Li, W. Duan, M. Ji, Y. Yang, S. Wang, and B. Liu, "The distributed permutation flowshop scheduling problem with different transport timetables and loading capacities," 2016 IEEE Congress on Evolutionary Computation, Vancouver, Canada.

[7] H. C. Hwang, and S.Y. Chang, "Parallel machines scheduling with machine shutdowns," Computers & Mathematics with Applications, Vol. 36, Iss. 3, pp. 21–31, August 1998.

[8] H. Kellerer, "Algorithms for multiprocessor scheduling with machine release times," IIE transactions, Vol. 30, Iss. 11, pp 991–999, November 1998.

[9] S. Hatami, R. R. García, and C. A. Romano, "The Distributed Assembly Parallel Machine Scheduling Problem with eligibility constraint," International Journal of Production Management and Engineering, vol. 3, pp. 13–23, January 2015.

[10] S. W. Lin, K. C. Ying, and C. Y. Huang, "Minimising makespan in distributed permutation flowshops using a modified iterated greedy algorithm", International Journal of Production Research, Vol. 51, pp 5029–5038, August 2013.

[11] B. Naderi, and R. Ruiz, "A scatter search algorithm for the distributed permutation flowshop scheduling problem," European Journal of Operational Research, Vol.239, Iss. 2, pp. 323–334, December 2014.

[12] S. Y. Wang, and L. Wang, "An estimation of distribution algorithm-based memetic algorithm for the distributed assembly permutation flow-shop scheduling problem," Systems, Man, and Cybernetics: Systems, IEEE Transactions. Vol.46, pp.139–149, April 2015.

[13] M. Hauschild, and M. Pelikan, "An introduction and survey of estimation of distribution algorithms," Swarm and Evolutionary Computation, Vol. 1, Iss. 3, pp. 111–128, September 2011.

[14] R. Armañanzas, I. Inza, R. Santana, Y. Saeys, et al. "A review of estimation of distribution algorithms in bioinformatics", BioData mining, Vol. 1, pp. 1-12, September 2008

[15] J. M. Framinan, and R. Leisten, "An efficient constructive heuristic for flowtime minimisation in permutation flow shops," Omega, Vol. 31, Iss. 4, pp. 311–317, August 2003.

[16] J. Ceberio, E. Irurozki, A. Mendiburu, and J. A. Lozano, "A distance-based ranking model estimation of distribution algorithm for the flowshop scheduling problem," IEEE Transactions on Evolutionary Computation, Vol.18, Iss. 2, pp. 286 – 300, April 2014

[17] P. Hansen, N. Mladenović, and J. A. M. Pérez, "Variable neighbourhood search: methods and applications," Annals of Operations Research, Vol. 175, Iss. 1, pp. 367–407, March 2010.

[18] S.Hatami, R.Ruiz, and C.Andrés-Romano, "Heuristics and metaheuristics for the distributed assembly permutation flowshop scheduling problem with sequence dependent setup times," International Journal of Production Economics, Vol. 169, pp. 76–88, November 2015.

[19] B. Jarboui, M. Eddaly, and P. Siarry, "An estimation of distribution algorithm for minimizing the total flowtime in permutation flowshop scheduling problems," Computers & Operations Research, Vol. 36, Iss.9 pp. 2638–2646, September 2009.

[20] S. Y. Wang, L. Wang, M. Liu, and Y. Xu, "An enhanced estimation of distribution algorithm for solving hybrid flow-shop scheduling problem with identical parallel machines," International Journal of Advanced Manufacturing Technology, Vol. 68, Iss.9, pp. 2043–2056, October 2013.

[21] L. Wang, S. Y. Wang, Y. Xu, G. Zhou, and M. Liu, "A bi-population based estimation of distribution algorithm for the flexible job-shop scheduling problem," Computers & Industrial Engineering, Vol. 62, Iss. 4, pp. 917–926, May 2012.

[22] S. -W. Lin, and K. –C. Ying, "A hybrid approach for single-machine tardiness problems with sequencedependent setup times," Journal of the Operational Research Society, Vol. 59, pp. 1109–1119, August 2008.

[23] H. B. Liu, A. Abraham, O. Choi, and S. H. Moon. "Variable neighborhood particle swarm optimization for multi-objective flexible job-shop scheduling problems," Lecture notes in computer science, Vol. 4247, pp. 197–204, October 2006.

[24] G. I. Zobolas, C. D. Tarantilis, and G. Ioannou, "Minimizing makespan in permutation flow shop scheduling problems using a hybrid metaheuristic algorithm," Computers & Operations Research, Vol.36, Iss.4, pp. 1249–1267, April 2009.