J Intell Manuf (2015) 26:861–871
DOI 10.1007/s10845-013-0821-3

A hybrid estimation of distribution algorithm for the semiconductor final testing scheduling problem

Shengyao Wang · Ling Wang · Min Liu · Ye Xu

Received: 29 May 2013 / Accepted: 2 August 2013 / Published online: 13 August 2013

Abstract As the last process of the semiconductor fabrication, the final testing is crucial to guarantee the quality of the integrated circuit products. The semiconductor final testing scheduling problem (SFTSP) is of great significance to the efficiency of the semiconductor companies. To find satisfactory solutions within reasonable computational time, the intelligent manufacturing scheduling based on the meta-heuristic methods has become a common approach. In this paper, a hybrid estimation of distribution algorithm (HEDA) is proposed to solve the SFTSP. First, novel encoding and decoding methods are proposed to map from the solution space to the schedule space effectively. Second, a probability model that describes the distribution of the solution space is built to generate the new individuals of the population. Third, a mechanism is used to update the parameters of the probability model with the superior solutions at every generation. Furthermore, to enhance the exploitation ability of the algorithm, a local search procedure is hybridized to find neighbor solutions of the promising individuals obtained by sampling the probability model. In addition, the influence of parameters is investigated based on Taguchi method of design-of-experiment, and a set of suitable parameters is suggested. Finally, numerical simulation based on some benchmark instances is carried out. The comparisons between the HEDA and some existing algorithms demonstrate the effectiveness of the proposed HEDA in solving the SFTSP.

Keywords Semiconductor final testing scheduling problem · Estimation of distribution algorithm · Probability model · Local search · Design-of-experiment

S. Wang (✉) · L. Wang · M. Liu · Y. Xu
Tsinghua National Laboratory for Information Science and Technology (TNList), Department of Automation, Tsinghua University, Beijing 100084, China
e-mail: wangshengyao@tsinghua.org.cn

Introduction

The semiconductor manufacturing process consists of two phases: the front-end process, i.e., the wafer fabrication, and the back-end process, i.e., the final testing. The final testing process is used to test the integrated circuit (IC) products after the completion of package and to check whether the products meet the requirements of standard specifications (Zhang et al. 2011). Due to their high prices, the testing machines of the semiconductor testing factories are not many. Therefore, it is very important to make an effective scheduling with an acceptable makespan (Jeng and Tsai 2010). In addition, as a specific type of simultaneous multiple resources scheduling problem, the SFTSP falls into the NP-hard category (Hao et al. 2013). Hence, the study of the SFTSP in theory and methodology has significant importance in both academic field and engineering field.

As a specific example of simultaneous multiple resources scheduling problem, the SFTSP is one of the most complex and common scheduling problems (Wu and Chien 2008). In the study of Uzsoy et al. (1991), the operations in the semiconductor testing facility were characterized by a broad product mix, variable lot sizes and yields, long and variable setup times, and limited test equipment capacity, then a suitable scheduling system was developed accordingly. Uzsoy et al. (1992) used the single-machine scheduling problems with sequence-dependent setup times as the research object to minimize both the maximum lateness with dynamic arrivals and the number of tardy jobs with a heuristic solution. Uzsoy et al. (1993) evaluated the performance of several dispatching rules in a semiconductor testing environment and examined the effects of uncertainties in problem data and job arrival patterns. The results of the simulation experiments provided suggestions to select the appropriate dispatching rules. Ovacik and Uzsoy (1996) presented the

decomposition methods to solve the SFTSP by exploiting the structure of the routings in semiconductor testing to decompose the shop into a number of work centers, which were scheduled using specialized procedures. Freed and Leachman (1999) described the multi-head tester scheduling problem and presented an enumeration solution procedure, which considered efficiency differences of using different number of heads. Based on the model of Uzsoy et al. (1991), Lin et al. (2004) used the theory of constraints to develop a heuristic capacity-constrained scheduling algorithm for the SFTSP. In recent years, some meta-heuristic algorithms have been applied for this problem. Wu and Chien (2008) developed a mathematical programming model to optimize the testing job scheduling and proposed a genetic algorithm (GA) to solve the problem in a short time for practical viability. Wu et al. (2012) developed a bi-vector encoding GA (bvGA) with a novel bi-vector encoding method, which represented the chromosomes of operation sequence and seizing rules for resource assignment, respectively. Hao et al. (2013) presented a cooperative estimation of distribution algorithm (CEDA) to incorporate a cooperative co-evolutionary paradigm to extend the model features and improve the evaluation lead-time. The CEDA had better performance than those of the GA (Wu and Chien 2008) and the bvGA (Wu et al. 2012), which was the best algorithm for solving the SFTSP reported in literature.

As a population-based algorithm, estimation of distribution algorithm (EDA) (Larranaga and Lozano 2002) has gained increasing attention and wide applications during recent years. According to the complexity of the model, the EDA can be classified as univariate model, bivariate model or multivariate model. The population-based incremental learning (Baluja 1994), univariate marginal distribution algorithm (Mühlenbein and Paass 1996) and compact GA (Harik et al. 1998) are univariate models, while mutual information maximization for input clustering (De Bonet et al. 1997), combining optimizers with mutual information trees (Baluja and Davies 1997) and bivariate marginal distribution algorithm (Pelikan and Mühlenbein 1999) are bivariate models. The factorized distribution algorithm (Mühlenbein and Mahnig 1999), the extended compact GA (Harik 1999) and the bayesian optimization algorithm (Pelikan et al. 1999) are multivariate models. For more details about the EDA, please refer to Larranaga and Lozano (2002).

The EDA is easy to implement with a parallel search framework, and it has few parameters to set. Due to its merits, the EDA has already been successfully applied to solve a variety of academic and engineering optimization problems, such as feature selection (Saeys et al. 2003), software testing (Sagarna and Lozano 2003), inexact graph matching (Cesar et al. 2005), flow-shop scheduling (Jarboui et al. 2009; Wang et al. 2013c), single machine scheduling (Chen and Chen 2013), resource-constrained project scheduling (Wang and Fang 2012), multi-dimensional knapsack problem (Wang et al. 2012a), flexible job-shop scheduling (Wang et al. 2012b, 2013a,b) and so on. In this paper, we will present a hybrid estimation of distribution algorithm (HEDA) to solve the SFTSP. We will propose novel encoding and decoding methods to map from the solution space to the schedule space effectively. Then, we will build a probability model that describes the distribution of the solution space to generate the new individuals of the population. We will also use a mechanism to update the parameters of the probability model with the superior solutions at every generation. To enhance the exploitation ability of the algorithm, we will hybridize the algorithm by a local search procedure to find neighbor solutions of the promising individuals obtained by sampling the probability model. Furthermore, we will investigate the influence of parameters based on Taguchi method of design-of-experiment (DOE), and suggest a set of suitable parameters. Finally, we will carry out the numerical simulation based on some benchmark instances. The comparisons between the HEDA and some existing algorithms will demonstrate the effectiveness of the proposed HEDA in solving the SFTSP.

The remainder of the paper is organized as follows: The "Problem description" section describes the SFTSP and the "Estimation of distribution algorithm" section introduces the basic EDA briefly. Then, the proposed HEDA for solving the SFTSP is introduced in details in the "HEDA for SFTSP" section. The influence of parameter setting is investigated based on design of experiment testing in the "Computational results and comparisons" section, and the computational results and comparisons are provided as well. Finally we end the paper with some conclusions in the last section.

Problem description

The semiconductor final testing is a complicated process, which can be described briefly as follows. There are n jobs (IC products) $J = \{J_1, J_2, \ldots, J_n\}$ to be tested on m machines $M = \{M_1, M_2, \ldots, M_m\}$. A job J_i is formed by a sequence of n_i operations $\{O_{i,1}, O_{i,2}, \ldots, O_{i,n_i}\}$ (e.g., functional test, burn-in, scan, bake, tape and reel, and package and load) to be performed one after another according to a given sequence. Note that, different jobs may require different steps of functional testing because of different product specifications. All jobs and machines are available at time 0. Preemption is not allowed, i.e., each operation must be completed without interruption once it starts. Compared to the classical job-shop scheduling problem, the SFTSP has the following additional properties.

1. Multi-resources. Besides the machine, more resources are needed, i.e., the tester, handler and accessory (Wu and

Chien 2008; Wu et al. 2012; Hao et al. 2013). A specific operation of the job can only be tested with appropriate configuration of resources. In addition, the resources of the SFTSP are so expensive that they can only be prepared in limited amounts (Pearn et al. 2004; Hao et al. 2013).

2. Flexible. There are $m_{i,j}$ machines $M_{i,j} \subseteq M$ that are capable for the execution of operation $O_{i,j}$, which indicates the virtual unrelated parallel machine environment (Chien and Chen 2007a,b). Therefore, the machine assignment is a sub-problem of the SFTSP.
3. Sequence-dependent setup times. Due to the setup activities including resource assembly and disassembly, temperature change, software download and calibration, a sequence-dependent setup time is further required to disassemble the original machine as well as assemble and calibrate the new machine for the incoming operation (Wu and Chien 2008; Hao et al. 2013). In addition, the setup is separable from process and is anticipatory so that a setup can start before the job is ready (Wu et al. 2012).

Therefore, the SFTSP can be regarded as multi-resources flexible job-shop scheduling with sequence-dependent setup times, which is illustrated in Fig. 1. The SFTSP is to determine both the assignment of machines and the sequence of operations on all the machines to minimize a certain scheduling objective function, e.g., the maximum completion time of all the jobs (makespan), without exceeding the total amount of available resources at any time. Because the mathematical model of the SFTSP (Wu and Chien 2008; Wu et al. 2012) has little relation with the design of the algorithm to be presented, it will not be covered in this paper.

Estimation of distribution algorithm

The estimation of distribution algorithms (EDA) (Larranaga and Lozano 2002) is a relatively new member of the family of evolutionary computing algorithms. The EDA employs explicit probability distributions to perform optimization procedure. It establishes a probability model of the most promising area by statistical information. Based on the probability model, it samples the search space to generate new individuals. To trace the more promising search area, it adjusts the probability model in each generation by using the information of some superior individuals of the new population. In Mühlenbein and Paass (1996), a general framework of the basic EDA was presented, as illustrated in Fig. 2.

To the EDA, it is the key issue to estimate the probability distribution. Since the probability model is use to describe the distribution of the solution among the space, it is crucial to build a proper probability model. In addition, the probability model should be well adjusted to make the search procedure tract the promising search area. As a consequence, a reasonable mechanism should be design to adjust the model. Since different problems have different properties, both probability model and the updating mechanism should be well adopted for a specific problem.

From the mechanism of the EDA, it can be seen that the EDA stresses more on exploration among the space while the exploitation capability within a certain area should be

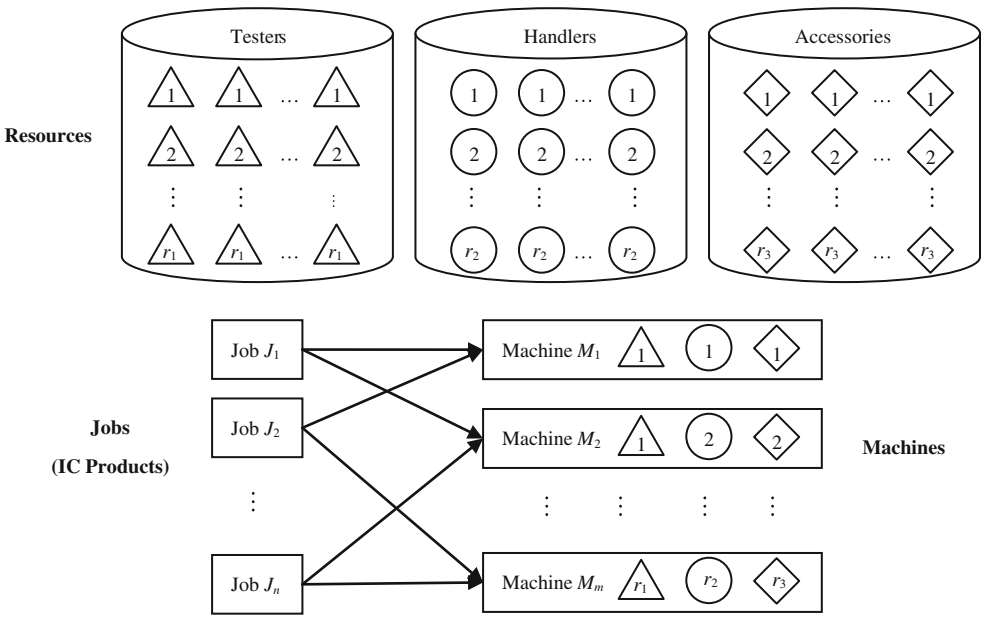

Fig. 1 Illustration of the SFTSP

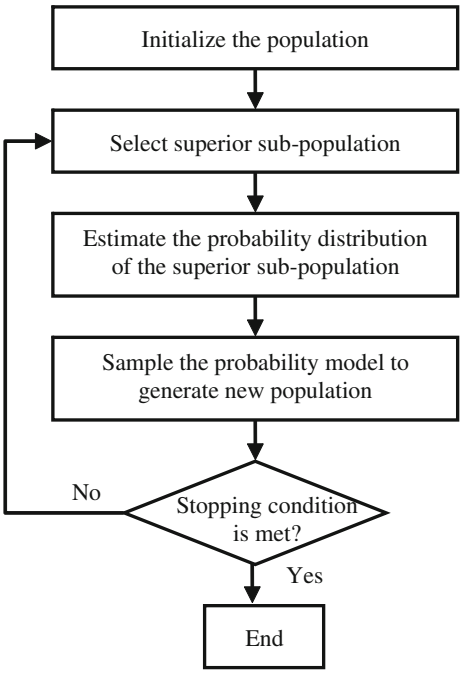

Fig. 2 The framework of the basic EDA

further enhanced. Well balancing the exploration and the exploitation abilities is crucial to develop a more effective EDA.

HEDA for SFTSP

In this section, we will propose a hybrid estimation of distribution algorithm (HEDA) to solve the SFTSP. First, we will introduce the solution representation, decoding method, probability model, updating mechanism, and local search strategy. Then, we will present the procedure of the HEDA and analyze the computation complexity.

Solution representation

To represent a solution of the SFTSP, both operation sequence and machine assignment should be determined. Thus, we encode a solution by using the processing sequence of operations and the assignment of operations on the machines. To be specific, we use two vectors to represent the solution, namely operation sequence vector and machine assignment vector, respectively. In the HEDA, the population contains a set of individuals represented by the two vectors.

The operation sequence vector represents processing order of all the operations. So, the number of genes of the operation sequence vector equals to the total number of all the operations T_O. We use the corresponding job number to denote the operation of a job. Thus, the kth occurrence of a job number in the vector refers to the kth operation in the processing sequence of the job.

The machine assignment vector represents the selection of machines for every operation. So, we use the corresponding machine number to denote the selected machine for processing a certain operation. Thus, the number of genes of the machine assignment vector is also T_O.

Considering a problem with 3 jobs / 3 machines, we provide an example to explain the above representation. As for the resource constraints, it is simply assumed that the testers, handlers and accessories are only enough to test two jobs at the same time. The processing data is shown in Tables 1 and 2. The representation of a feasible solution is illustrated in Fig. 3.

Decoding method

To decode a solution is to arrange the machines for the operations and determine the processing sequence on all the machines so as to form a feasible schedule. For a solution in the HEDA, the processing order of the corresponding schedule is in accordance with the operation sequence vector. To assign machines for the operations, the properties of the SFTSP, i.e., multi-resources and sequence-dependent setup time should be taken into consideration. For a given operation to schedule, find the corresponding machine according

Table 1 Processing time table

Operations	Machines		
	M_1	M_2	M_3
$O_{1,1}$	4	7	6
$O_{1,2}$	2	6	∞
$O_{2,1}$	4	5	7
$O_{2,2}$	3	∞	3
$O_{3,1}$	2	5	6
$O_{3,2}$	∞	3	2

Table 2 Setup time table

Machines	M_1	M_2	M_3
M_1	0	4	3
M_2	2	0	2
M_3	3	1	0

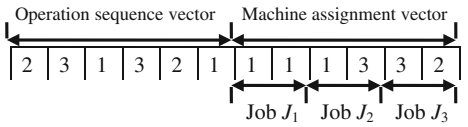

Fig. 3 Illustration of the representation of a feasible solution

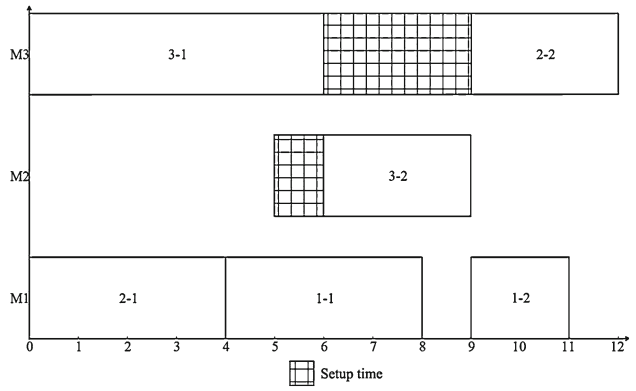

Fig. 4 Gantt chart of the solution shown in Fig. 3

to the machine assignment vector. Then, arrange it onto the machine at the earliest available time. That is to say, if there is no tester, handler or accessory for the machine configuration after the setup, it waits until all the resources are available.

From the representation shown in Fig. 3, we can get the detail operation sequence and machine assignment for this solution. That is, $(O_{2,1}, M_1)$, $(O_{3,1}, M_3)$, $(O_{1,1}, M_1)$, $(O_{3,2}, M_2)$, $(O_{2,2}, M_3)$, $(O_{1,2}, M_1)$. With the operation sequence and the machine assignment, we can further obtain the Gantt chart of this solution, as shown in Fig. 4. It can be seen that, the starting time of operation on machine M_1 is 9 instead of 8 because of the lack of resources at time 8.

Probability model

To well reflect the property of the considered problem, we use two probability matrixes to construct the probability model for the HEDA. The two matrixes are the operation probability matrix A_1 and the machine probability matrix A_2, which are used for operation sequence vector and machine assignment vector, respectively.

For the matrix A_1 at generation l, its element $p_{ij}(l)$ denotes the probability to place job j before or in position i of the operation sequence vector. To some extent, such a probability may refer to the significance to schedule the operation of a job on a certain machine. To uniformly sample the whole search area, the initial probability is initialized as $p_{ij}(l) = 1/n$ for all i $(i = 1, 2, \ldots, T_O)$ and j $(j = 1, 2, \ldots, n)$.

For the matrix A_2 at generation l, its element $q_{ijk}(l)$ denotes the probability to process operation $O_{i,j}$ on machine M_k. To some extent, such a probability may indicate the appropriateness of using a certain machine to process the operation. This matrix is initialized in a following way.

$$q_{ijk} = \begin{cases} \frac{1}{m_{i,j}}, & \text{if } O_{i,j} \text{ can be processed on machine } M_k \\ 0, & \text{else} \end{cases}, \forall i, j, k \tag{1}$$

The HEDA generates new individuals by sampling the probability model. Here, it samples the probability matrixes A_1 and A_2 to generate new individuals. To be specific, it first generates the operation sequence vector and then generates the machine assignment vector.

For the operation sequence vector, it selects job J_j with the probability of $p_{ij}(l)$ for every position i at generation l. If job J_j appears n_j times in the operation sequence vector, then the processing procedure of this job is finished. As a consequence, the jth column of A_1 will be set as zero. In a similar way, it generates the machine assignment vector to matrix A_2. Once both the operation sequence vector and the machine assignment vector are generated for an individual, the detailed schedule can be determined and the makespan can be calculated.

Updating mechanism

In addition to the sampling way based on the probability model, only by updating the probability model reasonably, the HEDA could trace the more promising search area. To suitably adjust the elements of the probability model by inheriting the good information of the population, the best SP solutions are selected as the superior sub-population. Then, we use the following equations to update the probability matrixes A_1 and A_2, respectively.

$$p_{ij}(l+1) = (1-\alpha)p_{ij}(l) + \frac{\alpha}{i \times SP}\sum_{s=1}^{SP} I_{ij}^s, \forall i, j \tag{2}$$

$$q_{ijk}(l+1) = (1-\beta)q_{ijk}(l) + \frac{\beta}{SP}\sum_{s=1}^{SP} \tilde{I}_{ijk}^s, \forall i, j, k \tag{3}$$

where $\alpha, \beta \in (0, 1)$ are the learning speeds of A_1 and A_2, respectively, I_{ij}^s and $\tilde{I}_{ijk}^s$ are the following indicator functions of the sth individual in the superior population.

$$I_{ij}^s = \begin{cases} 1, & \text{if job } j \text{ appears before or in position } i \\ 0, & \text{else} \end{cases} \tag{4}$$

$$\tilde{I}_{ijk}^s = \begin{cases} 1, & \text{if operation } O_{i,j} \text{ is processed on machine } M_k \\ 0, & \text{else} \end{cases} \tag{5}$$

Local search strategy

It is common recognized that incorporating local search into the framework of evolutionary computing can improve the optimization capability. As for the EDA, it pays more attention to the global exploration while the local exploitation needs to be further enhanced. In Wang et al. (2012a), a local search was incorporating into the EDA to develop a hybrid algorithm for well solving the multidimensional knapsack problem. In this paper, a simple method of the local search

strategy is designed to enhance the local exploitation to solve the SFTSP.

In particular, it randomly selects and changes one bit of the machine assignment vector of the best solution. If the new solution has a smaller makespan, it will replace the old solution. Such a procedure is applied on the best individual of the current population 10 times at every generation.

Procedure of the HEDA

With the design above, the procedure of the HEDA for solving the SFTSP is illustrated in Fig. 5.

After initializing the probability matrixes, the algorithm generates P individuals by sampling the probability matrixes. Then, the local search procedure is implemented on the best individuals and the best SP individuals are selected to update the probability model. If the stopping condition is met, the algorithm will stop and output the optimal solution; if not, the algorithm will generate P new individuals and start the next iteration. In this paper, the stopping condition is that the maximum number of evaluation times is reached.

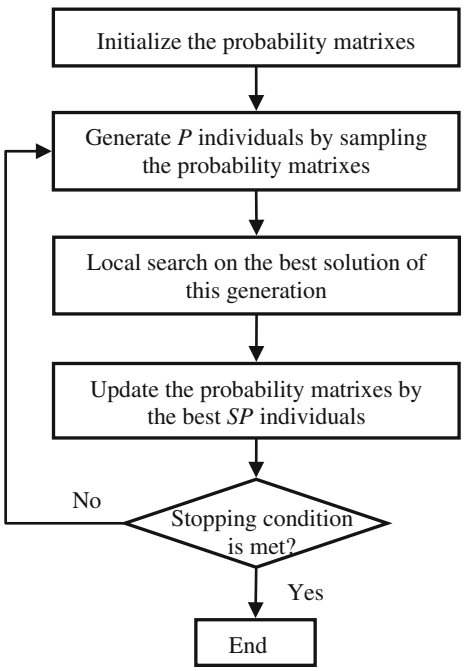

Fig. 5 The framework of the HEDA for the SFTSP

In summary, in the initial stage of evolution, the promising area of the solution space may be found by using the EDA-based estimating and sampling. Then, the local search procedure is performed in the "good" region to obtain better solutions. The benefits of the EDA and the local search are hybridized to balance global exploration and local exploitation.

Computation complexity analysis

For each generation of the designed HEDA, its computational complexity can be roughly analyzed as follow.

For the updating process, first it is with the computational complexity $O(P\log P)$ by using the quick sorting method to select the best SP individuals from population; then, it is with the complexity $O(T_o \times SP + T_o \times n)$ to update all the $T_o \times n$ elements of A_1 by the operator sequence vectors and with the computational complexity $O(T_o \times SP + T_o \times m)$ to update A_2 by the machine assignment vectors. Thus, the computational complexity for updating process is $O[T_o(SP + m + n) + P\log P]$.

For the sampling process, it generates a certain gene is by the roulette strategy via sampling A_1 and A_2 to obtain a new individual. It is with the complexity $O(T_o \times n)$ and $O(T_o \times m)$ to generate an certain operation sequence vector and machine assignment vector, respectively. Thus, the computational complexity for generating P individuals is $O[PT_o(m + n)]$.

From the above analysis, it can be concluded that the complexity of the proposed EDA is not large and the algorithm may have the potential capability to solve the SFTSP efficiently.

Computational results and comparisons

As for the evolutionary algorithms to solve the SFTSP, the wcGA (Wu and Chien 2008), the bvGA (Wu et al. 2012) and the CEDA (Hao et al. 2013) are three typical existing algorithms. In the literature, a set of simulation instances of the SFTSP (available at the Decision Analysis Lab website http://dalab.ie.nthu.edu.tw/newsen_content.php?id=0) was used for numerical testing. Thus, we also use these instances to carry out numerical simulations for testing and comparing the performance of the proposed HEDA with

Table 3 Parameters of the instances

Instance	n	m	Processing time	Setup time	Numbers of resources		
					Tester	Handler	Accessory
Large-scale (LS)	100	36	{1, 2, . . . , 15}	{1, 2, . . . , 5}	{10, 5, 3}	{10, 8, 4}	{7, 7, 5, 5}
Wide-range (WR)	60		{1, 2, . . . , 50}	{1, 2, . . . , 15}			

Table 4 Combinations of parameter values

Parameters	Factor level			
	1	2	3	4
P	50	100	150	200
SP	10	20	30	40
α	0.1	0.2	0.3	0.4
β	0.1	0.2	0.3	0.4

Table 5 Orthogonal array and RV values

Experiment number	Factor				RV
	P	SP	α	β	
1	1	1	1	1	123.05
2	1	2	2	2	111.25
3	1	3	3	3	107.10
4	1	4	4	4	124.25
5	2	1	2	3	107.55
6	2	2	1	4	104.80
7	2	3	4	1	131.65
8	2	4	3	2	119.35
9	3	1	3	4	109.80
10	3	2	4	3	104.40
11	3	3	1	2	119.85
12	3	4	2	1	141.60
13	4	1	4	2	125.35
14	4	2	3	1	141.55
15	4	3	2	4	108.65
16	4	4	1	3	117.40

the existing algorithms. The parameters of the instances are provided in Table 3. The algorithm is coded in C and run on a 2.3 GHz Intel Core i5 processor.

Parameters setting

The proposed HEDA contains several key parameters: the population size, i.e., P, the size of the superior subpopulation to update the probability model, i.e., SP, the learning speed of A_1, i.e., α, and the learning speed of A_2, i.e., β. To investigate the influence of these parameters on the performance of the algorithm, we implement the Taguchi method of design-of-experiment (DOE) (Montgomery 2005) by using the instance LS1. Combinations of different values of these parameters are listed in Table 4.

We set 4 factor levels for each parameter and choose the orthogonal array $L_{16}(4^4)$ according to the number of parameters and factor levels. That is, the total number of treatment is 16. The maximum number of evaluation times is set as 10,000. For each parameter combination, the HEDA is run 20 times independently and the response variable (RV) value is the average makespan value obtained by the HEDA in 20 times. Clearly, smaller value is the RV, better the combination is. The orthogonal array and the obtained RV values are listed in Table 5.

According to the orthogonal table, we illustrate the trend of each factor level in Fig. 6. Then, we figure out the average response value of each parameter to analyze the significance rank of each parameter. The results are listed in Table 6.

From Table 6, it can be seen that among the four parameters, the learning rate β of A_2 is the most significant one. It can be understood that, a proper learning rate of the machine assignment matrix is crucial to the performance of the HEDA. If the value of β is too large, the algorithm could suffer from the premature convergence to yield bad results. However, if the value is too small, slow convergence could take place, which will also result in bad results when the total computational budget is limited. Therefore, to ensure the effectiveness of the HEDA, an appropriate value of β is very essential. In

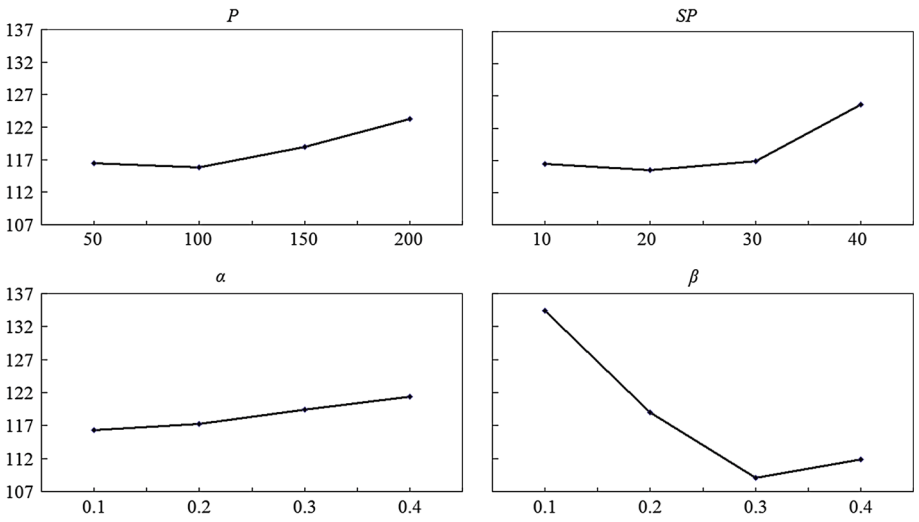

Fig. 6 Factor level trend of the HEDA

Table 6 Average response value

Level	*P*	*SP*	α	β
1	116.4125	116.4375	116.2750	134.4625
2	115.8375	115.5000	117.2625	118.9500
3	118.9125	116.8125	119.4500	109.1125
4	123.2375	125.6500	121.4125	111.8750
Delta	7.4000	10.1500	5.1375	25.3500
Rank	3	2	4	1

Table 7 Simulation results

Instance	wcGA		bvGA		CEDA		HEDA				Hypothesis test	
	Mean	SD	Mean	SD	Mean	SD	Best	Mean	SD	CPU(s)	*p* value	Sig($p < 0.05$)
LS1	127.33	1.42	124.20	1.72	121.83	1.39	102	**103.73**	**1.18**	4.19	0.00	Y
LS2	143.30	2.07	137.30	2.22	135.20	1.76	111	**112.90**	**1.37**	4.06	0.00	Y
LS3	135.27	3.57	131.13	1.94	123.60	1.99	101	**104.43**	**1.85**	3.89	0.00	Y
LS4	158.80	2.15	148.63	2.41	137.87	2.16	118	**121.43**	**1.85**	4.15	0.00	Y
LS5	143.97	2.02	142.47	1.84	138.23	1.67	112	**114.30**	**1.55**	4.10	0.00	Y
WR1	322.70	5.45	307.20	4.74	302.63	4.05	245	**251.07**	**3.74**	2.82	0.00	Y
WR2	248.37	5.65	240.53	3.69	234.83	3.47	202	**207.23**	**2.99**	2.35	0.00	Y
WR3	280.70	4.05	275.40	3.93	269.03	3.99	220	**225.33**	**3.95**	2.51	0.00	Y
WR4	229.23	4.57	220.20	3.97	209.30	4.56	185	**191.23**	**4.33**	2.36	0.00	Y
WR5	284.37	6.46	260.17	4.01	246.63	4.01	222	**229.27**	**3.92**	2.58	0.00	Y

The bold values mean the best results

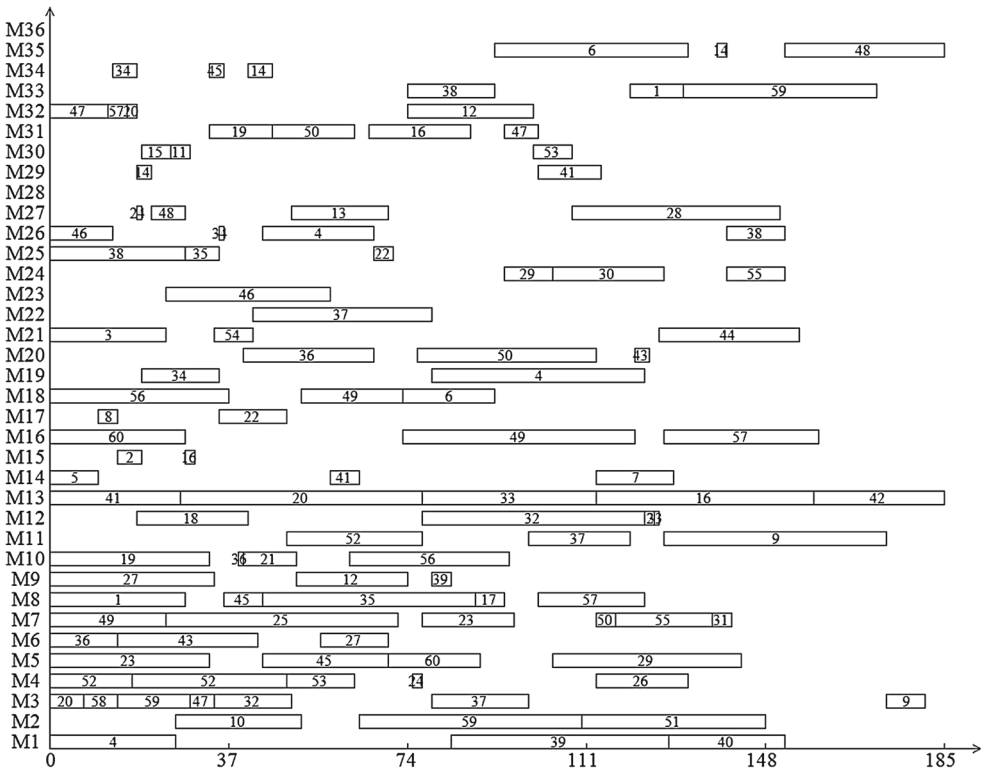

Fig. 7 Best solution of instance WR4 obtained by the HEDA

addition, the significant rank of the number of the selected superior individuals for updating the probability model, i.e. *SP*, is the second. A smaller value of *SP* can help to establish a more accurate model. Besides, the population size *P* ranks the third. A large population size ensures that the whole solution space can be sampled fully. However, in condition of a fixed maximum number of evaluation times, an over large population size leads to fewer generations so that the search may be not deep enough. Although the learning speed α of A_1 has the slightest effect to the HEDA, an appropriate value still can help the algorithm update the probability model accurately.

According to the above analysis, a better choice of the parameter combination is suggested as $P = 100, SP = 20, \alpha = 0.1$ and $\beta = 0.3$. This setting will also be used in the following comparison.

Simulation results and comparisons

To compare the HEDA with the wcGA (Wu and Chien 2008), the bvGA (Wu et al. 2012) and the CEDA (Hao et al. 2013), same as the literature, the instances are run 30 times independently. Besides, the evaluation times of the HEDA is 10,000 while the CEDA is 400,000 and the other two 800,000. It can be seen that the HEDA uses much less evaluations for solving the problem. Meanwhile, Hao et al. (2013) demonstrated that the CEDA was significantly better than the wcGA and bvGA. Thus, a hypothesis test is also performed to compare the HEDA with the CEDA. The results are listed in Table 7, where the results of the existing algorithms are directly from literature.

From Table 7, it can be seen the HEDA is significantly better than the other three algorithms for solving all the instances. The standard deviation (SD) values by the HEDA are the smallest, which demonstrates that the HEDA is the most robust one. Besides, the HEDA obtains smaller mean makespan values than the other three ones, which implies that the proposed HEDA is more effective for solving the SFTSP. In addition, Figs. 7 and 8 illustrate the Gantt chart and the remaining resources amount of the best solution obtained by the HEDA for instance WR4.

Besides, the average CPU times employed by the four algorithms are listed in Table 8.

From Table 8, it can be seen that the HEDA spends much less CPU time than the other three algorithms. The average CPU time of all the instances is only 3.3 s, which implies that the HEDA is very efficient for solving the SFTSP, even for the large-scaled problems.

From above comparisons, it can be concluded that the HEDA is efficient and more effective than the existing algorithms in solving the SFTSP. The superiority of the HEDA owes to the following aspects. (1) With the proper encoding and decoding method, it is helpful to obtain satisfactory schedules with makespan criterion. (2) With the well-designed probability model and the suitable updating mechanism, it is helpful to explore the search space effectively, especially within the promising area. (3) With local search procedure, it is helpful to improve the good schedule by enhancing the exploitation capability. With the above merits, the HEDA is more effective than the existing algorithms for solving the SFTSP.

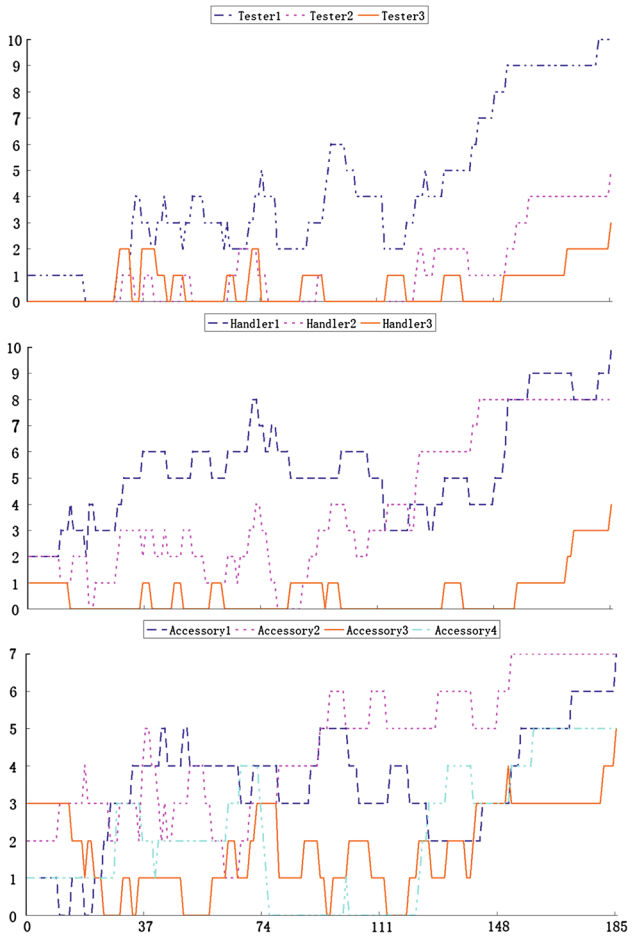

Fig. 8 The remaining resources amount of the solution in Fig. 7

Table 8 CPU time spent by the algorithms (s)

	wcGA[a]	bvGA[b]	CEDA[c]	HEDA[d]
LS instances	448.83	175.86	194.85	**4.08**
WR instances	179.82	94.93	105.87	**2.52**
Average	314.33	135.40	150.36	**3.30**

The bold values mean the best results
[a] Athlon 1.2 GHz CPU
[b] Core2 Duo 2.4 GHz CPU
[c] Core i5 2.8 GHz CPU
[d] Core i5 2.3 GHz CPU

Conclusions

In this paper, a hybrid estimation of distribution algorithm was proposed for solving the semiconductor final testing scheduling problem. The probability model-based EDA sampling and the local search procedure were combined to stress global exploration and local intensification together. We proposed novel encoding and decoding methods to map from the solution space to the schedule space effectively. We designed a probability model to describe the distribution of the solution space and generate the new individuals of the population. We used a mechanism to update the parameters of the probability model with the superior solutions. We hybridized the algorithm by a local search procedure to enhance the exploitation ability of the algorithm. Moreover, the influence of parameter setting was investigated by using DOE based testing. Simulation tests and comparisons to several existing algorithms demonstrated the effectiveness and efficiency of the proposed HEDA in solving the SFTSP. The future work is to design EDA for other types of scheduling problems, such as the lot-sizing scheduling problem.

Acknowledgments This research is partially supported by the National Key Basic Research and Development Program of China (No. 2013CB329503), the National Science Foundation of China (Nos. 61174189 and 61025018), the Doctoral Program Foundation of Institutions of Higher Education of China (No. 20100002110014), and the National Science and Technology Major Project of China (No. 2011ZX0250 4-008).

References

Baluja, S. (1994). *Population-based incremental learning: A method for integrating genetic search based function optimization and competitive learning*. Technical Report CMU-CS-94-163. Pittsburgh, PA: Carnegie Mellon University.

Baluja, S., & Davies, S. (1997). Using optimal dependency-trees for combinatorial optimization: Learning the structure of the search space. In *Proceedings of the 14th international conference on machine learning* (pp. 30–38).

Cesar, R. M., Bengoetxea, E., Bloch, I., & Larranaga, P. (2005). Inexact graph matching for model-based recognition: Evaluation and comparison of optimization algorithms. *Pattern Recognition*, *38*(11), 2099–2113.

Chen, S. H., & Chen, M. C. (2013). Addressing the advantages of using ensemble probabilistic models in estimation of distribution algorithms for scheduling problems. *International Journal of Production Economics*, *141*(1), 24–33.

Chien, C. F., & Chen, C. H. (2007a). A novel timetabling algorithm for a furnace process for semiconductor fabrication with constrained waiting and frequency-based setups. *OR Spectrum*, *29*(3), 391–419.

Chien, C. F., & Chen, C. H. (2007b). Using genetic algorithms (GA) and a colored timed Petri net (CTPN) for modeling the optimization-based schedule generator of a generic production scheduling system. *International Journal of Production Research*, *45*(8), 1763–1789.

De Bonet, J. S., Isbell, C. L., Jr. & Viola, P. (1997). MIMIC: Finding optima by estimating probability densities. In M. C. Mozer, M. I. Jordan, & T. Petsche (Eds.), *Advances in neural information processing systems* (pp. 424–430). Cambridge: MIT Press.

Freed, T., & Leachman, R. (1999). Scheduling semiconductor device test operations on multihead testers. *IEEE Transactions on Semiconductor Manufacturing*, *12*(4), 523–530.

Hao, X. C., Wu, J. Z., Chien, C. F., & Gen, M. (2013). The cooperative estimation of distribution algorithm: A novel approach for semiconductor final test scheduling problems. *Journal of Intelligent Manufacturing*. doi:10.1007/s10845-013-0746-x.

Harik, G. (1999). *Linkage learning via probabilistic modeling in the ECGA*. Illigal Report NO. 99010, Illinois Genetic Algorithms Lboratory, Illinois: University of Illinois at Urbana-Champaign.

Harik, G. R., Lobo, F. G., & Goldberg, D. E. (1998). The compact genetic algorithm. In *Proceedings of the IEEE conference on evolutionary computation* (pp. 523–528).

Jarboui, B., Eddaly, M., & Siarry, P. (2009). An estimation of distribution algorithm for minimizing the total flowtime in permutation flowshop scheduling problems. *Computer & Operations Research*, *36*(9), 2638–2646.

Jeng, W. D., & Tsai, M. S. (2010). Scheduling semiconductor final testing a DBR based simulation model. In *The 40th international conference on computers and industrial engineering* (pp. 1–6).

Larranaga, P., & Lozano, J. A. (2002). *Estimation of distribution algorithms: A new tool for evolutionary computation*. Netherlands: Springer.

Lin, J. T., Wang, F. K., & Lee, W. T. (2004). Capacity-constrained scheduling for a logic IC final test facility. *International Journal of Production Research*, *42*(1), 79–99.

Montgomery, D. C. (2005). *Design and analysis of experiments*. Arizona: Wiley.

Mühlenbein, H., & Mahnig, T. (1999). Convergence theory and applications of the factorized distribution algorithm. *Jounal of Computing and Information Technology*, *7*, 19–32.

Mühlenbein, H., & Paass, G. (1996). From recombination of genes to the estimation of distributions I: Binary parameters. *Lecture Notes in Computer Science*, *1141*, 178–187.

Ovacik, I. M., & Uzsoy, R. (1996). Decomposition methods for scheduling semiconductor testing facilities. *International Journal of Flexible Manufacturing Systems*, *8*(4), 357–388.

Pearn, W. L., Chung, S. H., Chen, A. Y., & Yang, M. H. (2004). A case study on the multistage IC final testing scheduling problem with reentry. *International Journal of Production Economics*, *88*(3), 257–267.

Pelikan, M., Goldberg, D. E., & Cantú-Paz, E. (1999). BOA: The bayesian optimization algorithm. In *Proceedings of the genetic and evolutionary computation* (pp. 525–532), San Francisco.

Pelikan, M., & Mühlenbein, H. (1999). The bivariate marginal distribution algorithm. In J. M. Benítez (Ed.), *Advances in soft computing: Engineering design and manufacturing* (pp. 521–535). London: Springer.

Saeys, Y., Degroeve, S., Aeyels, D., Van de Peer, Y., & Rouze, P. (2003). Fast feature selection using a simple estimation of distribution algorithm: A case study on splice site prediction. *Bioinformatics*, *19*(suppl 2), 179–188.

Sagarna, R., & Lozano, J. (2003). *On the performance of estimation of distribution algorithms applied to software testing*. Technical Report EHU-KZAA-IK-1/03. The University of the Basque Country, Spain.

Uzsoy, R., Church, L. K., Ovacik, I. M., & Hinchman, J. (1993). Performance evaluation of dispatching rules for semiconductor testing operations. *Journal of Electronics Manufacturing*, *3*(2), 95–105.

Uzsoy, R., Lee, C. Y., & Martin-Vega, L. A. (1992). Scheduling semiconductor test operations: Minimizing maximum lateness and number of tardy jobs on a single machine. *Naval Research Logistics*, *39*(3), 369–388.

Uzsoy, R., Martin-Vega, L. A., Lee, C. Y., & Leonard, P. A. (1991). Production scheduling algorithms for a semiconductor test facility. *IEEE Transactions on Semiconductor Manufacturing*, *4*(4), 270–280.

Wang, L., & Fang, C. (2012). An effective estimation of distribution algorithm for the multi-mode resource-constrained project scheduling problem. *Computer & Operations Research*, *39*(2), 449–460.

Wang, L., Wang, S. Y., & Fang, C. (2012a). An effective hybrid EDA-based algorithm for solving multidimensional knapsack problem. *Expert Systems with Applications*, *39*(5), 5593–5599.

Wang, L., Wang, S. Y., & Liu, M. (2013a). A Pareto-based estimation of distribution algorithm for the multi-objective flexible job-shop scheduling problem. *International Journal of Production Research*, *51*(12), 3574–3592.

Wang, S. Y., Wang, L., Liu, M., & Xu, Y. (2013c). An enhanced estimation of distribution algorithm for solving hybrid flow-shop scheduling problem with identical parallel machines. *The International Journal of Advanced Manufacturing Technology*. doi:10.1007/s00170-013-4819-y.

Wang, S. Y., Wang, L., Xu, Y., & Liu, M. (2013b). An effective estimation of distribution algorithm for the flexible job-shop scheduling problem with fuzzy processing time. *International Journal of Production Research*, *51*(12), 3778–3793.

Wang, L., Wang, S. Y., Xu, Y., Zhou, G., & Liu, M. (2012b). A bi-population based estimation of distribution algorithm for the flexible job-shop scheduling problem. *Computers & Industrial Engineering*, *62*(4), 917–926.

Wu, J. Z., & Chien, C. F. (2008). Modeling semiconductor testing job scheduling and dynamic testing machine configuration. *Expert Systems with Applications*, *35*(1–2), 485–496.

Wu, J. Z., Hao, X. C., Chien, C. F., & Gen, M. (2012). A novel bi-vector encoding genetic algorithm for the simultaneous multiple resources scheduling problem. *Journal of Intelligent Manufacturing*, *23*(6), 2255–2270.

Zhang, Z. C., Zheng, L., Hou, F., & Li, N. (2011). Semiconductor final test scheduling with Sarsa (λ, k) algorithm. *European Journal of Operational Research*, *215*(2), 446–458.