Cooperative Estimation of Distribution Algorithm: A Novel Approach for semiconductor final test scheduling problems

Xin-Chang Hao1, Jei-Zheng Wu2, Chen-Fu Chien3, Mitsuo Gen4

1Graduate School of Information, Production and Systems, Waseda University, Kitakyushu-shi, Fukuoka
2Department of Business Administration, Soochow University, Taipei, Taiwan
3Department of Industrial Engineering and Engineering Management, National Tsing Hua University, Hsinchu 30013, Taiwan
4Fuzzy Logic Systems Institute (FLSI), Japan and Hanyang University, Korea

gen@flsi.or.jp

Abstract

During the past several years, there have been a significant number of researches conducted in the area of semiconductor final test scheduling problems (SFTSP). As specific example of simultaneous multiple resources scheduling problem (SMRSP), intelligent manufacturing planning and scheduling based on meta-heuristic methods, such as Genetic Algorithm (GA), Simulated Annealing (SA), and Particle Swarm Optimization (PSO), have become the common tools for finding satisfactory solutions within reasonable computational times in real settings. However, limited researches were aiming at analyze the effects of interdependent relations during group decision-making activities. Moreover for complex and large problems, local constraints and objectives from each managerial entity, and their contributions towards the global objectives cannot be effectively represented in a single model.

In this paper, we propose a novel Cooperative Estimation of Distribution Algorithm (CEDA) to overcome the challenges mentioned before. The CEDA is established based on divide-and-conquer strategy and a co-evolutionary framework. Considerable experiments have been conducted and the results confirmed that CEDA outperforms recent research results for scheduling problems in FMS (Flexible Manufacturing Systems).

Keywords: Manufacturing management, Flexible manufacturing systems, Cooperative Estimation of Distribution Algorithm, semiconductor final test scheduling problems.

1. Introduction

Nowadays, most of high-tech industries including semiconductor, TFT-LCD, and solar cell are becoming increasingly flexible to face the highly dynamic supply chains and production networks. One critical issue in the industry is to improve capital effectiveness via managing capital expenditures (CapEx) under uncertain demand in terms of resources acquisition and capacity installation that can hardly attain instantly [1]. Meanwhile, Flexible Manufacturing Systems (FMS) allow the dynamic configuration of resources to process distinct products. It is possible to assign each of these products to more than one type of unrelated resources with different efficiencies. Multi-purpose resources could perform a wide range of tasks, which allows schedulers to concentrate the workloads amongst these resources in order to improve utilization rate and reduction of resource needs. A negative effect though, could be the reduction of production effectiveness and efficiencies due to the time wasted whenever a machine performs changeover and configuration to accommodate for the next job. Thus, it is vital to arrange production schedule that simultaneously consider the values of multiple resources in order to response to production needs rapidly and effectively [2].

Many scheduling problems always fall into NP-hard problem [3]. Multiple Resources Scheduling Problem (MRSP) [4–7], herein is not only to determine a process plan for each part (i.e. the determination of resource configuration), but also deciding the production schedule for multiple jobs, which tells the production team how, when and in which sequence to effectively allocate these operations of jobs to suitable manufacturing resources. Moreover, this process plan and schedule should maintain operation feasibility while excel in all objectives considered. This challenge greatly increases the complexity of MRSP. During the past several years, there has been a significant amount of researches conducted on MRSP. Intelligent manufacturing planning and scheduling based on meta-heuristics, such as Genetic Algorithms (GAs) [8], Simulated Annealing (SA) [5], Particle Swarm Optimization (PSO) [10], have become common tools for finding satisfactory solutions within reasonable computational times in real settings. Recently, Wu and Chien proposed mixed-integer linear programming model for semiconductor final test job scheduling dynamic testing machine configuration (SFTSP) and developed assignment algorithm [4]. For addressing the scheduling problem of multiple resources in general, Wu et al. modeled simultaneous multiple resource scheduling problem and proposed bi-vector encoding genetic algorithm representing the chromosomes of operation sequence and seizing rules for resource assignment in tandem [9], Kim et al. developed a symbiotic evolutionary algorithm to solve integrated Flexible Manufacturing System scheduling problems [11]. In their approach, the Evolutionary Algorithm (EA) coped with asymmetric multileveled structures, including machine allocation, tool allocation, alternative processes, and operation sequences in level 1 (the lowest level); loading and routing plans in level 2; loading/routing and sequencing in level 3; and overall system integration in level 4. This approach argued that parallel search approach always outperforms a single sequence search.

Although previous researches showed that the traditional EAs is a powerful optimization tool, there are however three primary reasons that suggest the algorithm is not entirely adequate for solving MRSP problems. Firstly, the population of individuals evolved by these algorithms has a strong bias to convergence in response to an increasing number of trials being allocated to observed regions of the solution space with above average fitness [12]. Secondly, for MRSP, there is a strong interdependent relationship between the resources configuration (process plan) of jobs and the schedule sequence of jobs. Unfortunately, this type of interaction among decision variables is inadequately considered in traditional EAs. Finally, for complex and large problems, local constraints and objectives from each managerial entity cannot be effectively represented in a single EA model.
This paper presents a Cooperative Estimation of Distribution Algorithm (CEDA) to solve the MRSP. CEDA incorporates cooperative co-evolutionary paradigm to extend the model features and improve evaluation lead-time of EDA. In general, the overall problem is split to a set of small-problems with respect to actual separation of management responsibilities. Such the decision space of small problem is represented as sub-populations within local evolutionary context (named species) following the population-based schema. Sub-population is evolved locally but at the same time cooperates with the other sub-populations at predefined intervals to ensure the solution searching direction aspires according to both local and global objectives. For each species, the probability model is employed to model the decision variables for estimating the joint distribution of multinomial data in order to generate new solutions. The proposed algorithm is not only capable of maintaining searching diversity in the evolution, but also the combination of information from the set of promising sub-solutions and the overall fitness information about the problem is used to estimate the distribution of the overall promising solutions.

The remainder of this paper is organized as follows: Section 2 gives an overview and mathematical model of the MRSP. In Section 3, the proposed CEDA approach is described and discussed in detail. Experiment comparisons that apply the CEDA approach for analyzing and solving the SFTSP problem is explained in Section 4. Finally, the paper is summarized and concluded in Section 5.

2. Simultaneous multiple resources scheduling problems (SMRSPs)

General SMRSPs are one of the most complex and common seen scheduling problems [4]. In SMRSPs, multiple resources are simultaneously combined and aligned to process a job. Without loss of generality, a set of jobs \( j \in J \) are ready to be scheduled for a short-term planning horizon. Each job is inseparable so that the process characteristics can be preserved for quality assurance and yield enhancement. Furthermore, each job has to be processed through a set of operations \( o \in O \). Normally, after an operation starts processing on a particular resource, it has to continue until completion, which is a non-preemption constraint.

Resource types can vary and include dedicated physical equipment, platforms, or interfaces between equipment and goods, and supporting labor. There are \( N \) types of heterogeneous resources \( R^o = \{r^1, ..., r^n\} \). \( R^r = \{r^n_1, ..., r^n_2\} \) for machine configuration, each of which contains homogeneous but unrelated resources, i.e. with different performances. Some of them can be very expensive and thus only be prepared in limited amount, say \( Q \), \( n \in \{1, 2, ..., N\} \), according to long-term capacity planning [13]. Thus, the sum of operations overlapping in time demanding any given resource should not exceed the total amount available. Multiple factors, such as maintenance, repairs, breakdowns, and ongoing R&D experiments can influence the availability of resources [14]. This study considers static resource availability whereas further research should take stochastic unavailability and dynamic release of multiple resources into account [15][16].

On the other hand, machine configuration is dynamic. Different operations may require different machine (configuration) types for processes, whereas more than one machine type can process one operation, which reveals the virtual unrelated parallel machine environment (Chien and Chen, 2007). Different machine types, \( m = (j^1, ..., j^n) \in M_\theta, r^n \in R^n, n \in \{1, 2, ..., N\} \), with dissimilar processing time, say \( P_m \), can perform an operation \( o \). However, it is not possible to configure resources arbitrarily to form a machine type. Consequently, the universal set of all machine types is \( M = \bigcup_{o \in \theta} M_\theta \subseteq R^n \times R^n \). To disassemble the original machine type \( l \) and to assemble and calibrate the new machine type \( m \) for the incoming operation, sequence-dependent setup time (SDST), denoted by \( S_{lm} \), is further required [4]. Setup activities comprise resources assembly and disassembly, temperature change, software download, and calibration. The setup is separable from process and is anticipatory so that a setup can start before a job is ready [1]. The reconfiguration decision is centralized control rather than hierarchical one that is interacted with multiple local and autonomous controllers [17]. A poor plan on machine configuration may waste resource availability on redundant setup or, on the other hand, lead to increasing unmet demand.

In particular, the semiconductor wafer probing and final test scheduling problems is specific examples of SMRSPs, which have common characteristics of SMRSPs include operation non-preemption, dynamic machine configuration with SDST, limited resources, and flexible job-shop. We developed a cooperative evolutionary algorithm to solve the MRSP followed by an efficient algorithm able to solve all characteristics in our previous work [9].

3. Cooperative Estimation of Distribution Algorithm

In the last decades, there has been a growing interest in optimization methods that explicitly identify and model the good solution found in previous generation, furthermore use the constructed model to guide the further search; these approaches are referred to Estimation of Distribution Algorithm (EDA). Population Based Incremental Learning (PBIL) algorithm is an optimization algorithm of EDA family [18]. The PBIL algorithm is combination of evolutionary algorithm (EA) and competitive learning, in PBIL, real vector (probability vector) presenting probability model of decision variables instead of individual member, and a simple incremental rule is used to update this vector after performing the selection on candidate solutions.

On the other hand, most stochastic optimization algorithms such as traditional EAs suffer from the curse of the dimensionality, which implies their performance deteriorates as the dimensionality of the search space increases. Potter and De Jong firstly introduces cooperative co-evolutionary paradigm to overcomes this disadvantages [12]. The basic idea of cooperative co-evolution is to divide a solution containing all of decision variables into many subcomponents, each subcomponent is represented by a corresponding EA model (named species), and evolve the species one by one (named fine-grain model) or concurrently, the fitness function is evaluated by combining the reprehensive from other species This approach restricts local parameters, variables and objectives to their respective subcomponents, which greatly reduces the complexity of exploiting the search space. They found that this decomposition achieves a significant performance over the basic EAs. Their approach just spitted the solution vector into small vectors sometime without considering the characteristics of problem. Fortunately, real life problems often consist of multiple sub-problems divided by the actual separation of management responsibilities.

3.1 Overview of CEDA

Fig. 1 presents cooperation scheme of our proposed CEDA incorporating the co-evolutionary paradigm. Firstly, each species \( d \) is created respectively and initializes probability vectors \( P_0 \) of its population with equal probability value. Before sampling
alternative solution from $\hat{P}_j(0)$ in population of species, for each probability vectors, search context (named partnership) which includes the cooperators representatives of others species should be created as search point, cooperation of the species is responsible for selecting representatives from species. Now, co-evolutionary context of CEDA is initialized. Thereafter, alternative solutions $S(0)$ are sampled from the partnership and given to cooperation component as feedback. Cooperation component collections feedback from the species completely, selects better solutions from alternative solutions and related probability vector $\hat{B}(0)$ as promising data; each species updates probability vector towards the best solution $\hat{B}(0)$ respectively; the iteration will continue until the predefined termination criteria is met. The pseudo-code for CEDA is presented in Fig. 2.

### 3.2 Subcomponents in CEDA

The following subsection presents components of the proposed CEDA in a more detailed way. Firstly introduction of cooperation mechanism is presented. Then, the representation of probability vector and learning mechanism is described. Finally, formulation of CEDA is summarized

#### (a) Representative selection for cooperation

In CEDA architecture, each probability vector in a species only represents a partition of the overall probability vector for decision variables. The partnership is use to cooperate with other species in order to determine the fitness of the probability vector within the ecosystem. The representative selection from species is vital function affecting the convergence and diversity of co-evolutionary. There has two approaches was discussed in Tan et al’s research work. The first is exhaustive approach; it cooperates with all individuals from other species, however, for large-scale problem, it suffers from computational complexity. It is reasonably practicable to cooperate with only certain members of other species to estimate its cooperation extend and fitness. Tan et al proposed another approach, in which the best two individuals in a species as representative set of the species [19]. However, it is often too greedy to selection the best individual solution as representative and results in premature potentially. For representative diversity, in our approach, $k$-tournament strategy (frequently $k = 2$) is adopted, unlike the greedy-strategy (best individual is always selected), $k$ individuals are randomly picked up from the top half of the best individuals in species, then the best individual from tournament with the highest fitness is selected as representative.

#### (b) Competitive learning of Probability vector

For traditional EAs, the representation of chromosome for each individual is generated by mapping the decision space into search space or directly encoding the decision variables, that is, the individual can be decoded to alternative solutions [20]. Meanwhile, PBIL adopts the probability vector for decision variables instead of individual member, which explicitly maintain the statistic contained in an EA’s population [21]. Each position in probability vector indicates the distribution of probability with respect to each variable. Generally, for discrete variables $X$, the domain of $X$ is set of predefined values $(x)$, in PBIL, No prior knowledge of distribution is assumed, and the distribution of random variable $X$ have the same equal probability, the initialization is presented below:

$$P_{g=0}(X) = \frac{1}{|X|}$$  \hspace{1cm} (1)

Where $|X|$ denotes the number of values in set of domain $X$. After the initialization of CEDA, probability vector in each species samples a new alternative solution, and the new solution is evaluated according to gene system objective. CEDA collects all the new alternative solutions and replace $\text{prSize} \times \text{elimRate}$ worse solutions kept in promising data. The probability distribution of $X$ can be estimated as following:

$$B_g(X = x) = \frac{N(X = x) + 1/|X|}{\text{prSize} + 1/|X|}$$  \hspace{1cm} (2)

**procedure:** Cooperation Estimation of Distribution Algorithm

**define:**

g: number of current generations.
d: index of specie.
$D$: number of species.
pop($d$): population of species $d$.
prSize: number of the promising solutions kept by CEDA.
elimRate: elimination rate of the keeping promising solutions.
stThreshold: stagnation threshold at which the partnership will be recreated.

**begin**

**Initialization:**
g←0:

**Step 1:** Create and initialize probability vector $\hat{P}_i(g)$ in the population pop($d$), $d \in [1..D]$.

**Step 2:** Create the partnership $\hat{P}_i(g)$ for each individual in pop($d$) by cooperation component, $d \in [1..D]$.

**Coevolution:**

**Repeat**

**Step 3:** Sample alternative solution $S(g)$ from the partnership for each probability vector in pop($q$), $q \in [1..Q]$.

**Step 4:** Cooperation component collects the alternative solution and related probability vectors and replace $\text{prSize} \times \text{elimRate}$ worse items with better solutions from $S(g)$ for each probability vector in pop($d$), $d \in [1..D]$ do

**Step 5:** Update the probability vector towards the best sample probability vector $\hat{B}_i(g)$.

**Step 6:** Perform mutation on probability vector to keep the diversity of sampling.

**end**

**Step 7:** when the partnership stagnates over stThreshold, it will be recreated by cooperation component.

g←g+1;

**until** terminating criterion is met.

**end**
Cooperative Estimation of Distribution Algorithm: A Novel Approach for Semiconductor Final Test Scheduling Problems

Where \( N(X = x) \) denotes the number of instances in promising solutions with variable \( X = x \); and \( 1/|X| \) gives the low bound to the probability of \( X \).

The distribution probability of \( X \) in probability vector is learned towards estimated distribution of promising data as following:

\[
P_g(x) = (1 - \alpha)P_g(x) + \alpha B(X = x)
\]

Where \( \alpha \) denotes the learning rate from the current promising solutions, in particular, for \( \alpha = 1 \), the probability distribution is reconstructed by current promising solutions completely.

In order to keep the diversity of sampling, after the distribution probability of \( X \) is updated towards the estimation distribution. Thereafter, the distribution can be tuned with probability \( p_m \) the mutation is performed by using the following definition:

\[
P_g(x) = \sum_{x' \in X \setminus \{x\}} \max(P_g(x) - \lambda_m / (|X| - 1, \varepsilon), (P_g(x) + \lambda_m))
\]

Where \( \lambda_m \) is the mutation shift that controls the amount for mutation operation, and \( \varepsilon \) is small probability-value in order to avoid the negative probability-value.

(e) Formulation of the CEDA

As depicted in the Fig. 2, in CEDA, each species starts to construct the probability vector and learns distribution probability. The learning pressure of network greatly depends on the size of promising solutions (prSize) of a species and elimination rate of the keeping promising solutions (elimRate) setting. After the construction of the network, CEDA samples the new alternative solutions, exploring the promising new areas of the solution searching space. The convergence rate of CEDA is largely determined by the number of sampling and prior individuals in species. For larger number of new individual samplings, the parameter of probability vector is biased to change frequently and increase the chance of premature convergence, because population diversity of the search space to be explored is lost. On the other hand, the parameters of the probability vector is reconstructed again when CEDA goes to the next evolutionary iterator, trade-off between the prior parameters and new parameters should be taken into account, which is represented by coefficient parameter \( a \).

4. Experiments and Discussion

4.1 Application of CEDA in SMRSPs

SMRSPs can be defined as the following decision making process: Machine assignment: determine the machine facility of operation which can be similarly processed on one of the multiple machine types candidates; machine setup planning: determine the accessory resources setup planning for operation according to the feature geometry and the available machining resources.

Following the schema of CEDA, SMRSPs can be mapped into two type species. One type is job species, of which number equals the number of jobs. The species corresponds to one job respectively, these species exploit machine assignment of related job respectively, they aim to construct probability model for machine assignment. The other type species (machine configuration species) exploits the scheduling problem including all of jobs. Machine configuration species aims to exploit the probability model of dispatch rules on each machine types, this study adopted five common dispatching rules, including random (RN), earliest completion time (ECT), shortest process time (SPT), shortest startup or setup time (SSST), and shortest setup plus process time (SSST). In order to reduce changeovers of the resources among the machine configurations, seizing rule-based heuristic algorithm is adopted to treat with resource loading and unloading, which enhanced computation by avoiding high-level complexity due to matching with resources and operations in our previous work [9].

<table>
<thead>
<tr>
<th>Problem</th>
<th># of Jobs</th>
<th># of Operations</th>
<th>Processing time</th>
<th>Setup time</th>
<th># of configurations</th>
<th># of Resource Types</th>
<th># of resource</th>
</tr>
</thead>
<tbody>
<tr>
<td>Single-Problem(S)</td>
<td>15</td>
<td>1</td>
<td>{1, ..., 6}</td>
<td>{1,2}</td>
<td>{1,2}</td>
<td>(R^1, R^2)</td>
<td>(R^1: {1,2,3}, ) (R^2: {1,2,3})</td>
</tr>
<tr>
<td>large-Scale(LS)</td>
<td>100</td>
<td>3</td>
<td>{1, ..., 15}</td>
<td>{1,2,3}</td>
<td>{1,2,3}</td>
<td>(R^1, R^2, R^3)</td>
<td>(R^1: {10,5,3}, ) (R^1: {10,8,4}, ) (R^1: {7,5,5}, ) (R^1: {10,5,3}, ) (R^1: {10,8,4}, ) (R^1: {7,5,5})</td>
</tr>
<tr>
<td>Wide-Range(WR)</td>
<td>60</td>
<td>3</td>
<td>{1, ..., 50}</td>
<td>{1,2,3}</td>
<td>{1,2,3}</td>
<td>(R^1, R^2, R^3)</td>
<td>(R^1: {10,8,4}, ) (R^1: {7,5,5})</td>
</tr>
</tbody>
</table>

\(R^1:\) tester, \(R^2:\) handler, \(R^3:\) accessory

| Table 2 Parameters and strategies of wcGA, bvGA and CEDA |
|-----------------------------|-----------------------------|-----------------------------|
| Iteration                   | 4000                        | 4000                        | 4000                                       |
| Population                  | 200                         | 200                         | 50                                         |
| Selection                   | tournament(k)               | tournament(k)               | -                                          |
| Strategy                    | elitism                     | elitism                     | -                                          |
| Operators                   | Crossover(P_m)              | Mutation(P_m)               | Sampling                                   |
| Parameters                  | P_m = 0.30                  | P_m = 0.30                  | prSize = 100                               |
|                            | P_c = 0.65                  | P_c = 0.65                  | stThresHold = 20                           |
|                            | k = 2                       | k = 2                       | a = 0.5                                    |
|                            | P_m = 0.4                   | k = 2                       | P_m = 0.4 k = 2                            |
4.2 Experimental setup

To examine the practical viability and efficiency of proposed CEDA, we designed a numerical study for comparison CEDA needed to be compared with efficient algorithms in the previous work. The proposed CEDA was compared with wcGA and bvGA based on a set of simulated data of semiconductor final testing scheduling problems. The data is available at Decision Analysis Lab (http://dalab.ie.nthu.edu.tw/newsen_content.php?id=0).
Parameters of simulation data were aligned in order to provide a comparative basis. The technical specification for the part is summaries on Table 1. The experiments were conducted on a personal computer with an Intel Core i5 CPU at 2.8G and 2GB RAM. The parameters and strategies of related algorithms are categorized in Table 4 respectively.

The five larger problems and five wide-range problems compared CEDA performance with the existing meta-heuristic, wcGA and bvGA. Table 3 shows that CEDA algorithms performed better than wcGA, bvGA respectively in all the experiments. Fig. 3 and Fig. 4 illustrate box plots for experimental results of the LS and WR problems, respectively. CEDA outperformed wcGA and bvGA in most problem instantiations.

5. Conclusions

In this paper, we proposed Cooperative Estimation of Distribution Algorithm (CEDA) for overcoming the challenges of modeling and evaluating complexity of Simultaneous Multiple Resources Scheduling Problem (SMRSP). Firstly, CEDA applied divide-and-conquer strategy to divide overall problem into partitions and each partition is mapped to sub-population (species) following evolutionary scheme. A positive result would indicate that the CEDA approach is suitable for providing decision support in a dynamic operation environment, where management preferences are often changed in response to the real-time conditions of the manufacturing system.

In our future work, further experiments will be conducted to determine the accuracy of our proposed CEDA in response to variations among the species of the ecosystem. The partnership which includes the cooperators from each species as search point is involved into CEDA, the construction and abandonment of partnership is introduced briefly, the policy of abandonment and reconstruction should be more study. Furthermore we will like to extend CEDA for adapting to multiple objective optimizations.

Acknowledgments

This research is partly supported by KOFST Project 2010-2011 at Hanyang University.

References