Knowledge-Based Systems 84 (2015) 1–9

Contents lists available at ScienceDirect

Knowledge-Based Systems

journal homepage: www.elsevier.com/locate/knosys

A knowledge-based multi-agent evolutionary algorithm for semiconductor final testing scheduling problem

Shengyao Wang, Ling Wang *

Tsinghua National Laboratory for Information Science and Technology (TNList), Department of Automation, Tsinghua University, Beijing 100084, China

ARTICLE INFO

Article history:
Received 9 November 2014
Received in revised form 14 March 2015
Accepted 19 March 2015
Available online 27 March 2015

Keywords:
Semiconductor final testing scheduling problem
Multi-agent
Evolutionary algorithm
Mutual-learning
Competition
Knowledge base

ABSTRACT

The final testing process ensures the quality of the products in the semiconductor manufacturing factory. Scheduling for the final testing process is crucial to the economic efficiency of production. In this paper, an effective knowledge-based multi-agent evolutionary algorithm (KMEA) is proposed for solving the semiconductor final testing scheduling problem (SFTSP). In the KMEA, each agent is represented by a solution, which is a combination of the operation sequence vector and the machine assignment vector. A hybrid initialization mechanism is proposed to balance the diversity and the quality of the initial agents. In each iteration of evolution, the agents evolve by mutual-learning and competition based on the model of agent lattice. Moreover, a knowledge base is employed to store the useful information during the search process. The knowledge base is used to generate new agents in the competition phase. The computational complexity of the KMEA is analyzed, and the influence of parameter setting is also investigated. Finally, numerical simulation and comparisons are provided to demonstrate the effectiveness and efficiency of the KMEA in solving the test instances.

1. Introduction

To ensure the quality of integrated circuit (IC) products, the products are tested in the semiconductor final testing process after the completion of wafer fabrication. Usually, the testing machines are not many in a semiconductor testing factory because of their high prices. As an essential function in production management, an effective scheduling is important to improve the utilization of resources [1,2]. Thus, the semiconductor final testing scheduling problem (SFTSP) becomes an important issue to the semiconductor manufacturing companies. In addition, the SFTSP is a specific type of simultaneous multiple resources scheduling problem, which has been proved to be NP-hard [3]. Therefore, it is of significant importance to study the SFTSP in both academic field and engineering field, especially to develop effective and efficient solution algorithms.

As an effective technology of distributed artificial intelligence, the agent-based optimization has gained wide applications in the field of evolutionary computation during recent years. Although the evolutionary algorithms (EAs) have strength on complex optimization [4,5], the EAs based on multi-agent system have the ability of performing better than the classical EAs [6]. Zhong et al. [7] proposed a multi-agent genetic algorithm (MAGA) by integrating GA and multi-agent system for solving the global numerical optimization problem. In their MAGA, each agent represented a candidate solution of the problem to be solved, and all the agents were fixed on the points of a lattice. In each generation, the agents competed or cooperated with their neighbors to obtain new better solutions. Based on the same model of agent lattice [7], Liu et al. proposed a multi-agent evolutionary algorithm for the constraint satisfaction problems [8] and the combinatorial optimization problems [9], respectively. By combining agents and quantum-bit, Tao et al. [10] proposed a quantum multi-agent evolutionary algorithm for partner selection problems in a virtual enterprise. Zeng et al. [11] presented a multi-agent evolutionary algorithm for assembly sequence planning based on the geometry and assembly process information of product.

Inspired by the above successful applications of the multi-agent EAs, this paper aims to propose a knowledge-based multi-agent evolutionary algorithm (KMEA) for solving the SFTSP. The innovation in designing the algorithm can be summarized as follows: (1) A novel initialization mechanism is proposed to ensure the diversity and quality of the initial solutions; (2) Some effective search operators are designed to explore better solutions in the mutual-learning phase and competition phase; (3) A knowledge base is designed to store the useful information during the search process and to generate promising solutions in the competition phase. In addition, we analyze the computational complexity of the KMEA

* Corresponding author. Tel.: +86 10 62783125; fax: +86 10 62786911.
E-mail address: wangling@mail.tsinghua.edu.cn (L. Wang).

http://dx.doi.org/10.1016/j.knosys.2015.03.024
0950-7051/© 2015 Elsevier B.V. All rights reserved.

and investigate the influence of parameter setting. Numerical simulation and comparisons show that the KMEA is more effective and efficient than the state-of-the-art in solving the test instances.

The remainder of the paper is organized as follows: Section 2 presents the literature review about SFTSP and Section 3 describes the SFTSP briefly. Then, the KMEA for solving the SFTSP is introduced in Section 4. The influence of parameter setting is investigated in Section 5, and the computational results and comparisons are provided as well. Finally the paper ends with some conclusions in Section 6.

2. Literature review

Scheduling for the semiconductor final testing process is one of the most complex and common scheduling problems [12]. To develop a suitable scheduling system, Uzsoy et al. [13] divided the facility or job shop into a number of work-centers and then sequenced one work-center at a time. Besides, the authors used the disjunctive graph representation of the entire facility to capture interactions between work-centers. For the single-machine scheduling problems with sequence-dependent setup times in the semiconductor test operations, Uzsoy et al. [14] presented heuristics to minimize both the maximum lateness with dynamic arrivals and the number of tardy jobs. Later, Uzsoy et al. [15] 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 [16] characterized the testing facilities by different types of work centers, such as sequence-dependent setup times and parallel identical machines. The authors exploited the structure of the routings and scheduled different work centers by specialized procedures. Freed and Leachman [17] described the multi-head tester scheduling problem, and presented an enumeration solution procedure considering efficiency differences of using different number of heads. Based on the model in [13], Lin et al. [18] used the theory of constraints to develop a heuristic capacity-constrained scheduling algorithm. The algorithm [18] outperformed other dispatching rules for the committed volume performance in many different operational conditions. Pearn et al. [19] presented a case study taken from a final testing shop floor in an integrated circuit manufacturing factory.

In recent years, some intelligent optimization algorithms have been applied to the SFTSP. In [12], a mathematical programming model was presented to optimize the testing job scheduling, and an algorithm was developed to specify the machine configuration of each job and allocate specific resources. Furthermore, the authors proposed a GA to solve the problem in a short time for practical implementation. In [20], a bi-vector encoding GA was proposed, where each solution was represented by the operation sequence and seizing rules for resource assignment. In [3], a divide-and-conquer strategy was applied to divide the overall problem into partitions, where each partition was mapped to a sub-population. The authors presented a cooperative estimation of distribution algorithm (CEDA) with better performance than the GAs [12,20]. Very recently, Wang et al. [21] proposed a hybrid estimation of distribution algorithm (HEDA), and Zheng et al. [22] presented a novel fruit fly optimization algorithm (nFOA). However, to the best of our knowledge, there is no reported work about multi-agent based algorithms for solving the SFTSP.

3. Problem description

The SFTSP 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 $M = \{M_1, M_2, \ldots, M_m\}$. The testing process of job J_i is to perform a sequence of n_i operations $\{O_{i,1}, O_{i,2}, \ldots, O_{i,n_i}\}$ according to a given sequence. The operations include functional test, burn-in, scan, bake, tape and reel, package and load. The processing time of $O_{i,j}$ on machine M_k is t_{ijk}. Due to different product specifications, different jobs may require different steps of functional testing. In addition, all the jobs and machines are available at the initial time. Once an operation starts, it cannot be interrupted before its completion. Compared to the classical job-shop scheduling problem (JSP), the SFTSP has some additional features.

(1) Multi-resources to dispatch simultaneously. The testing process requires more resources, i.e., the tester, handler and accessory [3,12,20]. To execute a specific operation for testing a job, the machine needs an appropriate configuration of resources. Moreover, the resources are in limited amounts [3,19]. Therefore, the resource dispatching should be considered in the SFTSP.

(2) Flexibility of machine assignment. For the execution of operation $O_{i,j}$, a set of machines $M(i,j) \in M$ is capable, where $m_{ij} = |M(i,j)|$ denotes the number of the machines. It is flexible to assign machines for the operations, i.e., a virtual unrelated parallel machine environment [23,24]. Therefore, the machine assignment is one of the sub-problems in the SFTSP.

(3) Sequence-dependent setup times. To assemble and calibrate the new machine for the incoming operation, a sequence-dependent setup time is further required for the setup activities in the SFTSP [3,12]. In addition, the setup is anticipatory and separable from the testing process. Therefore, the setup on the new machine can start before the operation completes on the original machine [20].

In brief, the SFTSP is a multi-resources flexible job-shop scheduling problem with sequence-dependent setup times, which is illustrated in Fig. 1 [21].

The SFTSP is to determine both the assignment of machines and the sequence of operations on all the machines, so as to minimize the makespan without exceeding the total amount of available resources at any time. Therefore, it is much more complex than the classical JSP. For the mathematical model, please refer to [12,20].

4. Knowledge-based multi-agent evolutionary algorithm

In this section, the KMEA for solving the SFTSP is introduced in details. First, the flowchart of the KMEA is presented. Then, the solution representation, initialization mechanism, mutual-learning phase, and competition phase are introduced. Besides, the computational complexity of the KMEA is analyzed.

4.1. Flowchart of the KMEA

Knowledge base and agent lattice are two key components of the KMEA. The knowledge base is used to store the useful problem-specific information and to generate the new agents. The agent lattice provides a physical environment for the agents. The model of the agent lattice [7] is illustrated in Fig. 2. With the lattice model, the agents perform effective search operators by the interaction.

From Fig. 2, it can be seen that the size of the agent lattice is L. Thus, the total number of the agents is $L \times L$. All the agents are fixed on the lattice-points to interact with their neighbors [7]. Each agent denotes a solution of the SFTSP. The objective is to find an optimal solution through the iterative search procedure.