sustainability

MDPI

Article

Simulation Study for Semiconductor Manufacturing System: Dispatching Policies for a Wafer Test Facility

Hyun Joong Yoon [1] and Junjae Chae [2,*]

[1] School of Mechanical and Automotive Engineering, Daegu Catholic University, Geongsan-si, Gyeongbuk 38430, Korea; yoon@cu.ac.kr

[2] School of Air Transport, Transportation and Logistics, Korea Aerospace University, Goyang-si, Gyeonggi-do 10540, Korea

* Correspondence: jchae@kau.ac.kr

Received: 19 December 2018; Accepted: 14 February 2019; Published: 20 February 2019

Abstract: The manufacture of semiconductor products requires many dedicated steps, and these steps can be grouped into several major phases. One of the major steps found at the end of the wafer fabrication process is the electrical die sorting (EDS) test operation. This paper focuses on dispatching policies in an EDS test facility to reduce unnecessary work for the system. This allows the semiconductor manufacturing facility to achieve better overall efficiency, thereby contributing to sustainable manufacturing by reducing material movements, the use of testing machines, energy consumption, and so on. In the facility, wafer lots are processed on a series of workstations (cells), and the facility holds identical parallel machines. The wafers are moved by an automatic material handling system from cell to cell as well as within cells. We propose several scheduling policies consisting of intercell and intracell material movements for efficient system operation. For this, four intercell scheduling policies and two intracell scheduling policies are introduced, and the effects of combinations are tested and evaluated through simulation experiments to obtain performance measures such as cycle time and work in process. The most efficient results among the combinations are presented as a proposed scheduling policy for a given EDS test facility.

Keywords: dispatching policy; semiconductor; wafer probing; wafer sorting; sustainable system

1. Introduction

Integrated circuit (IC) manufacturing has become an industry-leading area in recent years, and it accordingly receives great attention from researchers. As the demand for production increases, environmental concerns are also elevated [1], and thus improving on environmental protection and sustainable practices while maintaining productivity are strategic goals for manufacturing firms today [2]. The IC manufacturing process in particular involves many automatic machining steps, and improving this machining process directly influences sustainability in the manufacturing system [3]. Thus, reducing the use of resources and optimizing the mostly automated system could be viewed as one of the goals necessary to work towards sustainable manufacturing.

In general, the IC manufacturing process is composed of the following four major phases: wafer fabrication, wafer probing (wafer sort), packaging (assembly), and the final test [4,5], as shown in Figure 1. Wafer fabrication is a highly complex technological process and highly capital-intensive [3–5]. In this process, the semiconductor devices are manufactured in a fab, and this is termed as front-end operations [6]. This requires a long cycle time and takes numerous steps in a clean environment [3,4,6]. The individual chips on the wafers are tested via electrical die sorting (EDS) for functionality at the end of the front-end operation phase. After this process, the wafers move to the assembly stage and final test, which are called back-end operations [6]. The chips are encapsulated in a plastic or ceramic material to protect them from contamination, and the devices are subsequently inspected to confirm whether they satisfy the required specifications [2,6]. In the EDS test, numerous operations should be employed to assess the reliability of the functioning chips on a wafer before that wafer is sent to the next stage. Several kinds of machines are used, and the material flow for the steps is either circular or linear in form. Various scheduling approaches can be used to efficiently job sequence this operation flow. The goals of this scheduling are to shorten the cycle time, increase the throughput, and lower the work in process (WIP). The proper allocation of resources combining the routing of material movements for scheduling would facilitate increased system efficiency.

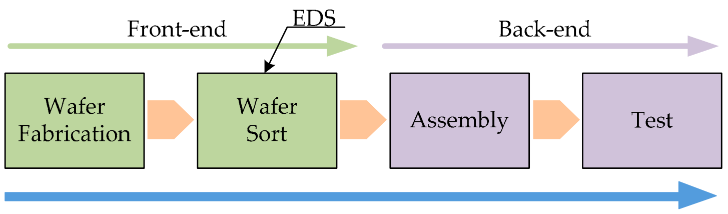


Figure 1. A semiconductor production process—a schematic view.

In this paper, we analyze system performance for the predefined layout of an EDS test facility in several terms, such as time in system and WIP. The layout of the facility includes several cells and stockers. The machines are grouped into a cell for their function or usability and identical machines could be allocated to a different cell for improved system performance. This sort of machine allocation influences the scheduling effect on system performance. The facility configuration in this paper is based on one currently in use. An automatic material handling system (AMHS) controls the front open unified pod (FOUP) intercell and intracell movements. We propose four scheduling policies for intercell schedulers and two for intracell schedulers. The system needs both inter- and intracell movements to complete the test process, and we selected four reasonable combinations of two kinds of operations and investigate their performance. The AutoMod®simulation tool is used to evaluate the performance of the proposed heuristic scheduling policies, and the results are compared to each combination to find the most favorable policy for a given facility. Figure 2 shows the sample layout of an EDS.

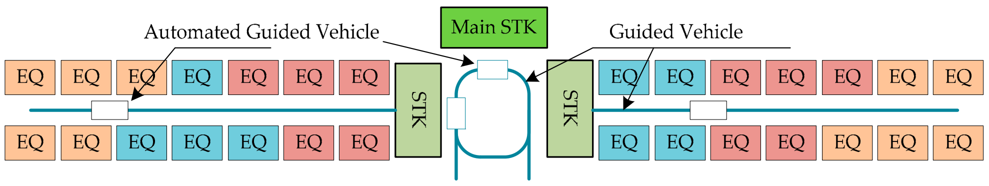


Figure 2. An electrical die sorting (EDS) sample layout.

The remainder of the paper is organized as follows. Section 2 provides a review of the related literature. In Section 3, the problem is described, and the basic representation of the problem is presented. The proposed heuristic scheduling policies are presented in detail in Section 4. Specific simulation experiments and experimental comparisons are explained in Section 5. Section 6 concludes the research, including a discussion of future work.

2. Literature Review

In semiconductor manufacturing, much research literature since the late 1980s has focused on the scheduling problems of wafer fabrication phase. Regarding the scheduling problems of semiconductor testing facilities, the first research report can be found in the paper by Uzsoy et al. [6] Uzsoy et al. [6] characterized the operations in the testing facility as a broad product mix, variable lot sizes and yields, long and variable setup times, and limited test equipment capacity. They first divided the facility, or job shop, into a number of work centers, and represent the job shop using a disjunctive graph to capture interactions between work centers. They insist that the proposed approach is able to efficiently obtain solutions to particular work center subproblem. Then Ovacik and Uzsoy [7] presented a shift bottleneck algorithm for scheduling of semiconductor testing facility that is composed of work centers. They show that the proposed shifting bottleneck algorithm performs better than dispatching rules by simulation experiments. Later Ovacik and Uzsoy [8] also presented a decomposition method for scheduling a semiconductor testing facility. They assume that the facility is characterized by different types of work centers, some of which have sequence-dependent setup times and some parallel identical machines. Chen et al. [9] presented a Lagrangian relaxation approach for the scheduling problems of a wafer test (or wafer probing or wafer sorting) and final test facility with preemptiveness. They formulated the scheduling problem as an integer programming problem and have solved the problem by using the Lagrangian relaxation to minimize the total weighted tardiness. Then, Chen and Hsia [10] presented a scheduling method for a wafer test and final test facilities with precedence constraints using Lagrangian relaxation technique.

Huang and Lin [11] present a human–computer interactive scheduler, named the interactive computer-aided scheduling system, to cope with the combinatorial difficulties of scheduling problems with sequence-dependent setup costs and multiple criteria for a wafer test facility. An experiment has been conducted to compare the performance of the proposed scheduling approach with six priority rules. De and Lee [12] presented a knowledge-based scheduling system for a semiconductor final test facility. They characterized the scheduling problem as a generalized job shop problem with both parallel workstation clusters and batch processors. The proposed scheduling system consists of a knowledge base developed using a frame-based knowledge representation scheme, and a solution strategy based on filtered beam search. Yang and Chang [13] address a multiobjective scheduling problem for minimizing cycle time and on time delivery in wafer test and final test facilities. The Pareto optimization approach is used and a new scheduling algorithm is proposed to solve the dual problem. Xiong and Zhou [14] proposed a Petri net-based hybrid heuristic search strategies to cope with the complexities for multiple lot scheduling for a semiconductor test facility. Sivakumar [15] proposed a simulation-based online near-real-time dynamic scheduling system to reduce cycle time in a final test facility. The proposed scheduling system has been implemented at a semiconductor back-end site. Lee et al. [16] addressed the wafer testing process, which requires several types of equipment with different capacities. They proposed LP-based heuristic algorithms for monthly production and capacity planning, and daily production planning and scheduling. Pearn et al. [17] presented a case study on the wafer test scheduling problem, where the jobs are clustered by their product types that are processed on groups of identical parallel machines. They formulated the wafer test scheduling problem, which is a kind of a classical parallel-machine scheduling problem, as an integer programming model to minimize the total machine workload. They demonstrated the applicability of the method using a real-world example in the paper by Pearn et al. [18] Then, Pearn et al. [19] addressed the wafer test scheduling problem with due date restriction. Pearn et al. [20] also presented a case study on the final test scheduling problem with reentry, which is a kind of a variation of the complex flow-shop scheduling problem. They proposed three network algorithms to solve the final test scheduling problem.

Chiang et al. [21] presented a scheduling method for semiconductor wafer test facility. They have modeled the wafer testing flow using colored-timed petri nets, which are used to simulate the production processes and keep track of the equipment status and lot conditions. Then they presented a genetic algorithm-based approach to solve the scheduling problem, with dispatching policies for lot and equipment selections. Chiang et al. [22] presented a rule-based scheduling method for a wafer test facility with three due date-base objectives including the tardy rate, total tardiness, and the maximum tardiness. Instead of legacy dispatching rule paradigm, they proposed a new paradigm to better utilize the dispatching rules, in which they use a genetic algorithm to generate appropriate dispatching rules depending on the system status and target performance criteria. Lin at al. [23]

presented a capacity-constrained scheduling method by using the concept of the theory of constraints for a semiconductor final test facility. The addressed scheduling problem is characterized by a broad product mix, variable lot sizes and yields, long and variable setup times, and limited test equipment capacity. Song et al. [24] addressed the scheduling problem in semiconductor assembly and final test facilities. They focus on the scheduling of the bottleneck stations, since many assembly and final test facilities are designed based on the theory of constraints. They first formulate the bottleneck station scheduling problem and then apply ant colony optimization technique to solve it metaheuristically. They also proposed an ant colony-based scheduling framework and provide the system parameter tuning. Pearn et al. [25] deal with a multistage wafer test scheduling problem with reentry, which is regarded as the identical parallel-machine scheduling problem. They present sequential and parallel strategies, where the former schedules the jobs at the required stages according to the sequence of manufacturing process and the latter is designed specifically for the reentrant characteristic.

Lee et al. [26] presented a daily planning and scheduling system for the wafer test facility. They proposed a practical mathematical formulation for the daily planning and heuristic procedure for the scheduling to minimize the test change-over within the daily target. Bang and Kim [4] addressed a scheduling problem for a semiconductor wafer test facility, where wafer lots are processed on a series of workstation with identical parallel machines. In order to minimize total tardiness, they presented a heuristic algorithm, where a bottleneck-focused scheduling approach was employed, and priority-rule-based algorithms were used for scheduling at the workstations including the bottleneck workstation. Lin et al. [27] addressed the scheduling problem of a wafer test facility. They presented three meta-heuristic algorithms— an ant colony system algorithm, a genetic algorithm, and a Tabu search algorithm—with total setup time minimization as the primary criterion and the minimization of the number of machines as the secondary criterion. With the identical scheduling problem definition in [27], Ying [28] presented a heuristic scheduling method based on the iterated greedy approach [29]. Doleschal et al. [30] addressed a scheduling problem of a wafer test facility, which is regarded as a single operation problem with unrelated parallel machines, release dates, setup, and dedications. They presented a scheduling approach, in which a static resource allocation problem is solved by mixed integer programming to remove unnecessary equipment allocations in the dedication scenario, and then feasible scheduling solutions are generated by a discrete event simulation based on the reduced dedication matrices.

Bard et al. [31] proposed a daily scheduling method of multipass lots in assembly and final test facilities with reentry. They formulated the scheduling problem as a mixed integer programming problem, and presented a multistage scheduling approach using a reactive greedy randomized search procedure. Pearn et al. [32] addressed the burn-in test scheduling problem that can be found in a semiconductor final test facility and thin film transistor liquid crystal display (TFT-LCD) manufacturing facility. The burn-in test scheduling problem, first proposed by Lee et al. [33], is regarded as a multidimensional parallel-batch processing machine scheduling problem. The authors formulate the burn-in test scheduling problem as a mixed integer linear programming problem and present two heuristic algorithms to minimize total setup and processing times in the burn-in test processes. Hao et al. [34] present a cooperative estimation of distribution algorithm to solve a semiconductor final test scheduling problem. The proposed method incorporates a cooperative coevolutionary paradigm to extend the model features and improve the evaluation lead-time of an estimation of distribution algorithms. Zheng et al. [35] proposed a novel fruit fly optimization algorithm to solve a semiconductor final testing scheduling problem, which is regarded as a simultaneous multiresource flexible job-shop scheduling problem with sequence-dependent setup times, with the objective of minimizing the maximum completion time of all the operations. Bard et al. [36] presented optimization and simulation approaches for semiconductor assembly and final test facilities. The optimization model is solved with a greedy randomized adaptive search procedure [31] that is designed to focus on hot lots while taking the cost of changeovers into account. Wang et al. [37] presented a hybrid estimation distribution algorithm for a semiconductor final testing scheduling problem. They combine the estimation of distribution algorithm with a local search procedure to enhance the exploitation ability of the algorithm. Kim et al. [38] present agent-based scheduling methods for hybrid cellular production line, which is one of the types of wafer probe centers in the semiconductor industry. They propose heuristic scheduling rules for both intercell and intracell scheduling problems.

Gao et al. [39] present a three-phase methodology for a scheduling assembly and final test operations for semiconductor devices. In the first phase, the authors solve the assignment of tooling and lots to machines, and in the second phase, the lots are optimally sequenced. Finally, the machines are reset to allow additional lots when tooling is available. The proposed methodology is tested using data provided by the assembly and test facility of a leading manufacture. Jia et al. [40] showed how the logic of intelligent heuristics can be combined with discrete event simulation software (AutoSched AP) to evaluate various dispatch rules for machine setup and scheduling in the assembly and final test facilities. Three new dispatch rules are presented to configure machines and assigning lots in the facilities. Hur et al. [41] investigated the setup scheduling problem at the semiconductor assembly and final test facilities in a multimachine and multitooling environment. The objective is to minimize the number of shortages of key devices and to maximize the weighted throughput over a two- to five-day planning horizon. They presented a hierarchical machine setup scheduling model to determine machine setups, lot assignments, and lot sequences using a greedy randomized adaptive search procedure. Jia et al. [42] statistically compared six dispatch rules for semiconductor assembly and final test facilities. The objectives are to minimize the weighted sum of key device shortages and to maximize the weighted throughput of lots processed. The statistical analyses show that the two rules designed to process as many hot lots as possible offer the best performance in terms of minimizing the weighted shortage.

Wang and Wang [43] presented a knowledge-based multiagent evolutionary algorithm to solve a semiconductor final testing scheduling problem. Each agent is represented by a solution that is a combination of the operation sequence vector and the machine assignment vector. The agents evolve via mutual learning and competition based on the agent lattice model. Joung et al. [44] presented a multipass oriented scheduling heuristic algorithm to maximize the throughput of semiconductor final test facilities. The algorithm determines test schedules, machine setups, and job assignments. Sang et al. [45] proposed a cooperative coevolutionary invasive weed optimization algorithm for a semiconductor final testing scheduling problem. The algorithm iterates with two coupled colonies, one of which addresses the machine assignment problem, while the other deals with the operation sequence problem.

3. Problem Description

This paper deals with the scheduling of the wafer test facility that is assumed to be a sort of hybrid cellular production line. A wafer test facility consists of cells and each cell is composed of more than one processing machine. There are two kinds of stockers to store jobs. Jobs (or FOUPs) start and end at a main stocker. The other kind of stocker is a cell stocker. Every cell has its own cell stocker to store jobs temporally in cells. There exists an intracell material handling system (Intra-MHS), such as an automated guided vehicle, in each cell to transfer jobs within a cell, and there is an intercell material handling system (Inter-MHS), such as an overhead transfer system, to transfer jobs from cell to cell. For example, a job is firstly released into a main stocker. If the job in the main stocker is allocated to one of cells, an Inter-MHS transfers the job to the cell stocker of the target cell. An Intra-MHS moves the job from the cell stocker to a machine and vice versa. When the whole operations that are to be processed in the cell are finished, the job waits in the cell stocker to be ordered to a next cell for the remaining operations. After completing the whole operations of the job, it goes back to the main stocker. In addition to the above described consideration, this study considers multiple types of jobs with identical operation sequences, which implies that although they have identical operation sequences their processing times may be different depending on the its job type. This is a general situation in a wafer test facility for semiconductor memory.

Selected nomenclature is defined as follows.

- J is the finite *set of jobs*.
- C is the finite *set of cells*.
- M is the finite *set of machines*.
- $M^c \subset M$ is the finite set of machines in the cell $c \in C$.
- M_T is the finite *set of machine types*.
- $(O_j, \prec)$ is the finite *sequence of operations* of job $j \in J$.
- $(P_j, \prec)$ is the *partition of operation sequence* O_j.
- $\eta : M \to M_T$ is the *machine-to-machine-type function*.
- $\delta : O_j \to M_T$ is the *operation-to-machine-type function*.
- $\chi : P_j \to C$ or $\chi : O_j \to C$ is the *cell allocation function*.
- $\psi : O_j \to M$ is the *machine allocation function*.

With respect to the partition of the operation sequence, for instance, let us assume that job j has five operations: $O_j = < o_{j1}, o_{j2}, o_{j3}, o_{j4}, o_{j5} >$. Then, one of the possible partitions of operation sequence O_j is $P_j = < p_{j1}, p_{j2}, p_{j3} > = << o_{j1}, o_{j2} >, < o_{j3} >, < o_{j4}, o_{j5} > >$. The machine-to-machine-type function $\eta(m)$ returns the machine type ($\in M_T$) of machine $m \in M$. The operation-to-machine-type function $\delta\left(o_{jk}\right)$ returns the machine type ($\in M_T$), where operation $o_{jk} \in O_j$ should be processed. The cell allocation function $\chi(p_{jp} \in P_j)$ allocates each element of P_j to proper cell. For instance, $\chi(p_{j1} =< o_{j1}, o_{j2} >) = c_1$, $\chi(p_{j2} =< o_{j3} >) = c_2$, and $\chi(p_{j3} =< o_{j4}, o_{j5} >) = c_1$ imply that $< o_{j1}, o_{j2} >$, $< o_{j3} >$, and $< o_{j4}, o_{j5} >$ are processed in cell c_1, c_2, and c_1, where $c_1,\ c_2 \in C$, respectively. The machine allocation function $\psi\left(o_{jk}\right)$ returns the machine, in which operation $o_{jk} \in O_j$ should be processed. Note that the sequences of operations, the operation-to-machine-type functions, and the machine-to-machine-type functions should be given in advance, whereas the partitions of operation sequences, the cell allocation functions, and the machine allocation functions should be determined by the scheduler.

In addition, time related indices are defined as follows.

- $t_r(j)$ is the *released time* of job $j \in J$.
- $t_q(j)$ is the *completion time* of job $j \in J$.
- $t_{cs}(p_{jp})$ is the *cell starting time* of $p_{jp} \in P_j$ in cell $\chi(p_{jp})$.
- $t_{cf}(p_{jp})$ is the *cell finishing time* of $p_{jp} \in P_j$ in $\chi(p_{jp})$.
- $t_s\left(o_{jk}\right)$ is the *starting time* of $o_{jk} \in O_j$ in machine $\psi\left(o_{jk}\right)$.
- $t_f\left(o_{jk}\right)$ is the *finishing time* of $o_{jk} \in O_j$ in $\psi\left(o_{jk}\right)$.
- $T_{cp}(p_{jp})$ is the *cell processing time* of $p_{jp} \in P_j$ in $\chi(p_{jp})$.
- $T_p\left(o_{jk}\right)$ is the *processing time* of $o_{jk} \in O_j$ in $\psi\left(o_{jk}\right)$.
- T_{cm} is the moving time from cell to cell by an Inter-MHS.
- T_m is the moving time in a cell by an Intra-MHS.

The released time $t_r(j)$ can be defined as the time when job j is released into the main stocker or the time when job j in a main stocker is allocated to the first cell. In this paper, the second definition is used. The completion time $t_q(j)$ is the time when job j comes back into the main stocker after completing all its operations. The cell starting time $t_{cs}(p_{jp})$ is defined as the time when job j arrives at cell $\chi(p_{jp})$ to perform its operation(s) in $p_{jp} \in P_j$, and the cell finishing time $t_{cf}(p_{jp})$ is the time when job j returns to the stocker of $\chi(p_{jp})$ after finishing whole operation(s) in p_{jp}. The starting time $t_s\left(o_{jk}\right)$ is defined as the time when operation o_{jk} starts its process in machine $\psi\left(o_{jk}\right)$, and the finishing time $t_f\left(o_{jk}\right)$ is defined as the time when operation o_{jk} is finished in $\psi\left(o_{jk}\right)$. The finishing time $t_f\left(o_{jk}\right)$ can be simply computed by $t_s\left(o_{jk}\right) + T_p\left(o_{jk}\right)$. The cell processing time $T_{cp}(p_{jp})$ is the time required by p_{jp} to be processed in $\chi(p_{jp})$. The cell processing time includes waiting times in both a stocker and machines, processing times in machines, and moving times by an Intra-MHS. The processing time $T_p\left(o_{jk}\right)$ is the time required by o_{jk} to be processed in $\psi\left(o_{jk}\right)$.

The intercell scheduling problem is to find a proper partition of operation sequence P_j for every job in J, and to determine the cell allocation function $\chi(p_{jp})$ for every $p_{jp} \in P_j$ so that a given objective function is minimized. For instance, the objective function to minimize mean cycle time is given by

$$\text{Minimize } \frac{1}{|J|}\sum_{j=1}^{|J|}\left[t_q(j) - t_r(j)\right]. \tag{1}$$

The completion time of job $j \in J$ is obtained as follows. Let $p_{j|P_j|}$ is the last element of P_j, i.e., $p_{jp} \prec p_{j|P_j|}\ \forall p_{jp} \in P_j$. Then, the completion time is computed by $t_q(j) = t_{cf}\left(p_{j|P_j|}\right) + T_{cm}$.

Constraints:

- If $o_{ja} \neq o_{jb}$ for $\forall o_{ja}, o_{jb} \in O_j$, then $\delta(o_{ja}) \neq \delta\left(o_{jb}\right)$.
- If $\exists p_{jp} \in P_j,\ \chi(p_{jp}) = c \in C$, then $\exists o_{jk} \in p_{jp} : \#(m \in M_c) > 0$ s.t. $\delta\left(o_{jk}\right) = \eta(m)$.
- If $p_{jp}^{-} \neq \varnothing,\ t_{cs}(p_{jp}) \geq t_{cf}\left(p_{jp}^{-}\right) + T_{cm}$; Otherwise, $t_{cs}(p_{jp}) \geq t_r(j) + T_{cm},\ \forall j \in J,\ \forall p_{jp} \in P_j$.
- $t_{cf}(p_{jp}) = t_{cs}(p_{jp}) + T_{cp}(p_{jp}),\ \forall j \in J,\ \forall p_{jp} \in P_j$.

The fist constraint implies that every operation of a job should be processed in a different kind of machine type. The second constraint states that there should exist at least one machine in $\chi(p_{jp})$ for every operation in p_{jp}. The third constraint ensures the precedence condition of operations for every job, in which p_{jp}^{-} is the precedent element of p_{jp}, i.e., $p_{jp}^{-} \prec p_{jp}$ and there exists no p_{jp^*} such that $p_{jp}^{-} \prec p_{jp^*}$ and $p_{jp^*} \prec p_{jp}$. The last constraint is to compute the cell finishing time $t_{cf}(p_{jp})$, which is sum of the cell starting time and the cell processing time in $\chi(p_{jp})$.

The intrascheduling problem in cell c^* is to determine the machine allocation function $\psi\left(o_{jk}\right)$ for every $o_{jk} \in p_{jp}$ such that $\chi(p_{jp}) = c^*$. For instance, the objective function can be defined as minimization of the total tardiness of all jobs allocated to cell c^*. The tardiness of p_{jp} in $\chi(p_{jp})$ is the maximum value between zero and the cell finishing time, $t_{cf}(p_{jp})$, minus its estimated value, $\hat{t}_{cf}(p_{jp})$. The estimated cell finishing time $\hat{t}_{cf}(p_{jp})$ is anticipated by the intercell scheduler, which will be further discussed in the following section. In this case, the objective function of the intracell scheduling problem is given by

$$\text{Minimize } \sum\nolimits_{\forall j \in J, \chi(\forall p_{jp} \in P_j)} = c^* \max\left[0, t_{cf}(p_{jp}) - \hat{t}_{cf}(p_{jp})\right]. \tag{2}$$

Constraints:

- If $o_{ja} \neq o_{jb}$ for $\forall o_{ja}, o_{jb} \in O_j$, then $\delta(o_{ja}) \neq \delta\left(o_{jb}\right),\ \forall j \in J,\ \forall p_{jp} \in P_j\ s.t.\ \chi(p_{jp}) = c^*, \forall o_{ja}, o_{jb} \in p_{jp}$.
- If $o_{jk}^{-} \neq \varnothing,\ t_s\left(o_{jk}\right) \geq t_s\left(o_{jk}^{-}\right) + T_p\left(o_{jk}^{-}\right) + T_m$; Otherwise, $t_s\left(o_{jk}\right) \geq t_{cs}(p_{jp}) + T_m,\ \forall j \in J, \forall p_{jp} \in P_j\ s.t.\ \chi(p_{jp}) = c^*, \forall o_{jk} \in p_{jp}$.
- $t_s\left(o_{jk}\right) + T_p\left(o_{jk}\right) \leq t_s(o_{il}) \vee t_s(o_{il}) + T_p(o_{il}) \leq t_s\left(o_{jk}\right),\ \forall i,j \in J,\ \forall p_{ip} \in P_i\ s.t.\ \chi(p_{ip}) = c^*, \forall p_{jp} \in P_j\ s.t.\ \chi(p_{jp}) = c^*, \forall o_{il} \in p_{ip}, \forall o_{jk} \in p_{jp},\ \psi(o_{il}) = \psi\left(o_{jk}\right)$.
- $t_{cf}(p_{jp}) = \max\left[t_s\left(o_{jk}\right) + T_p\left(o_{jk}\right)\right] + T_m,\ \forall j \in J,\ \forall p_{jp} \in P_j\ s.t.\ \chi(p_{jp}) = c^*, \forall o_{jk} \in p_{jp}$.

The first constraint of the intracell scheduling is same with that of the intercell scheduling. The second constraint ensures the precedence condition of operations for every job in a cell, where o_{jk}^{-} is the precedent element of o_{jk}, i.e., $o_{jk}^{-} \prec o_{jk}$ and there exists no o_{jk^*} such that $o_{jk}^{-} \prec o_{jk^*}$ and $o_{jk^*} \prec o_{jk}$. The third constraint states that only one operation should be processed in a machine at a time. The last constraint is to compute the cell finishing time.

4. Heuristic Scheduling Policies

4.1. Scheduling Policies for Nter-Cell Scheduler

Four heuristic scheduling policies are presented and investigated for the intercell scheduler. The role of the intercell scheduler is to determine a proper partition of operation sequence P_j and target cell $\chi(p_{jp})$ for every $p_{jp} \in P_j$.

4.1.1. Balanced Random Distribution

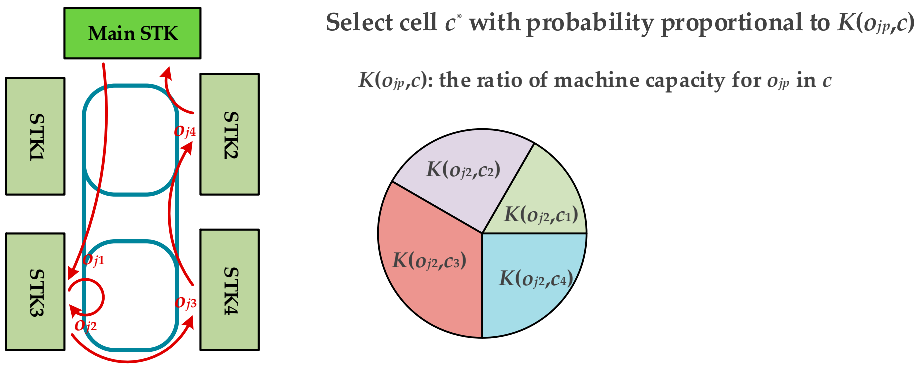
The balanced random distribution (BRD) is used to select a target cell for the job to be dispatched as shown in Figure 3. It randomly selects the target cell in proportion to the machine capacity ratio. If the next operation of the job is o_{jp}, the machine capacity ratio of cell c is defined as the number of machines in c whose type is $\delta(o_{jp})$ over the total number of machines whose type is $\delta(o_{jp})$. In the BRD rule, for every $j \in J$, the partition of operation sequence, P_j, is exactly identical with the operation sequence O_j, i.e., $p_{jp} = o_{jp}$, $\forall p_{jp} \in P_j$, $\forall o_{jp} \in O_j$. This implies that the BRD rule determines the target cell $\chi(o_{jp})$ for every single operation $o_{jp} \in O_j$ $(p = 1, 2, \ldots, |O_j|)$. This also means that there exists no machine-to-machine direct transfer in a cell. For instance, let us assume that job j arrives at $\chi(o_{jp})$ for operation o_{jp} to be processed. After finishing its operation, the job returns the cell stocker of the current cell and waits to be allocated to a next target cell by the intercell scheduler. Firstly, the BRD computes the ratio of machine capacity for every operation in each cell. The ratio of machine capacity of operation $o_{jp} \in O_j$ in cell $c \in C$ is defined as follows.

$$K(o_{jp}, c) = \frac{\text{Num. of machines in } c \text{ whose type is } \delta(o_{jp})}{\sum_{c \in C} \text{Num. of machines whose type is } \delta(o_{jp})}. \tag{3}$$

Then, the BRD rule randomly selects the target cell c^* for o_{jp} proportional to the ratio of machine capacities of candidate cells. The target cell c^* for o_{jp} $(= p_{jp})$ is determined by the following procedure.

```
for each $c \in C$
    compute $K\left(o_{jp}, c\right)$
for $i$ :=1 to $|C|$
    $cum_i \leftarrow \sum_{c=1}^{i} K\left(o_{jp}, c\right)$
$r \leftarrow$ Random $[0, 1]$
if $r < cum_1$
    $c^* \leftarrow 1$
else
    select $c^*$ such that $cum_{c^*-1} < r < cum_{c^*}$
```

In the above procedure, cum_i is the cumulative value of the ratio of machine capacities. The cell with the higher ratio of machine capacity is likely to be selected through the BRD rule.



Main STK
STK1
STK2
STK3
STK4
o_{j1}
o_{j2}
o_{j3}
o_{j4}
Select cell c^* with probability proportional to $K(o_{jp},c)$
$K(o_{jp},c)$: the ratio of machine capacity for o_{jp} in c
$K(o_{j2},c_2)$
$K(o_{j2},c_1)$
$K(o_{j2},c_3)$
$K(o_{j2},c_4)$

Figure 3. Balanced random distribution (BRD) policy. In selection of a target cell for o_{jp}, it considers the ratio of machine capacities with respect to o_{jp}. The cell with the higher machine capacity has a higher probability to be selected.

4.1.2. Shortest Stocker Queue Length

The shortest stocker queue length (SSQL) is used to select a target cell for the job to be dispatched. It selects the target cell with the shortest stocker queue length. More specifically, if the next operation of the job is o_{jp}, the SSQL computes the stocker queue length of every cell with respect to operation o_{jp}, and it then estimates the finishing time of o_{jp} in the corresponding cell. The SSQL selects the target cell with the earliest estimated finishing time of o_{jp}, or the shortest stocker queue length. In the SSQL rule, likely in the BRD rule, the partition of operation sequence, P_j, is identical with the operation sequence O_j, for every $j \in J$. The SSQL rule, as explained in Figure 4, for every candidate cell of o_{jp}, anticipates the finishing time for job j to complete its operation o_{jp} in machine type $\psi(o_{jp})$ by considering the total processing times of the jobs that are being processed or is waiting to be processed in $\psi(o_{jp})$, and the number of machines of $\psi(o_{jp})$. Then, the SSQL selects the target cell with the shortest anticipated cell finishing time. The target cell c^* for o_{jp} $(= p_{jp})$ is determined by the following procedure.

```
for each $c \in C$
  $M_p^c \leftarrow \left\{m \middle| m \in M^c \text{ and } \eta(m) = \delta\left(o_{jp}\right)\right\}$
  if $M_p^c \neq \varnothing$
    $\hat{Q}_p^c \leftarrow 0$
    for each $o_{ik}$ such that $\chi(o_{ik}) = c$ and $\delta(o_{ik}) = \delta\left(o_{jp}\right)$
      $\hat{Q}_p^c \leftarrow \hat{Q}_p^c + T_p(o_{ik})$
    $\hat{Q}_p^c \leftarrow \hat{Q}_p^c / \left|M_p^c\right|$
  else
    $\hat{Q}_p^c \leftarrow$ infinite
  $\hat{t}_{cf}^c\left(o_{jp}\right) \leftarrow \hat{Q}_p^c + T_p\left(o_{jp}\right) + T_{cm}$
select $c^*$ such that $\arg\min_c \hat{t}_{cf}^c\left(o_{jp}\right)$
```

In the above procedure, $\hat{Q}_p^c$ is the estimated queue length (or estimated waiting time) to start operation o_{jp}, and $\hat{t}_{cf}^c\left(o_{jp}\right)$ is the estimated cell finishing time of o_{jp} in c.

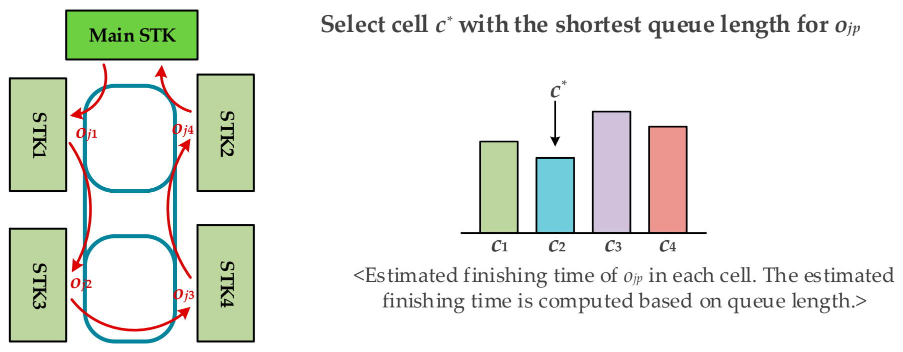

Main STK
STK1
STK2
STK3
STK4
o_{j1}
o_{j2}
o_{j3}
o_{j4}
Select cell c^* with the shortest queue length for o_{jp}
c^*
c_1
c_2
c_3
c_4
<Estimated finishing time of o_{jp} in each cell. The estimated finishing time is computed based on queue length.>

Figure 4. Shortest stocker queue length (SSQL) policy. In selection of a target cell for o_{jp}, it considers the queue length for o_{jp}. First, it computes the estimated finishing time of o_{jp} in each cell, which is computed based on its queue length. Then, the cell with the shortest estimated finishing time is selected.

4.1.3. M-SSQL (Modified-SSQL)

The M-SSQL is used to select a target cell for the job to be dispatched as shown in Figure 5. It is similar with the SSQL in that it selects the target cell with the shortest stocker queue length. However, in the M-SSQL, the current cell, in which the job is located currently, has higher priority. More specifically, if the next operation of the job is o_{jp}, the SSQL computes the stocker queue length of every cell with respect to operation o_{jp}, and it then estimates the finishing time of o_{jp} in the corresponding cell. Exceptionally, the estimated finishing time of the cell where the job is currently located is subtracted by a predefined advantage parameter. The M-SSQL then selects the target cell with the earliest estimated finishing time of o_{jp}. Like in the SSQL, there exists no machine-to-machine direct transfer in a cell in the M-SSQL rule. The target cell c^* for o_{jp} $\left(= p_{jp}\right)$ whose current cell is c^o is determined by the following procedure.

```
for each $c \in C$
    $M_p^c \leftarrow \left\{ m \middle| m \in M^c \text{ and } \eta(m) = \delta\left(o_{jp}\right) \right\}$
    if $M_p^c \neq \varnothing$
        $\hat{Q}_p^c \leftarrow 0$
        for each $o_{ik}$ such that $\chi(o_{ik}) = c$ and $\delta(o_{ik}) = \delta\left(o_{jp}\right)$
            $\hat{Q}_p^c \leftarrow \hat{Q}_p^c + T_p(o_{ik})$
        $\hat{Q}_p^c \leftarrow \hat{Q}_p^c / \left|M_p^c\right|$
    else
        $\hat{Q}_p^c \leftarrow$ infinite
    if $c = c^o$
        $\hat{Q}_p^c \leftarrow \hat{Q}_p^c - \mu_1$
    $\hat{f}_{cf}^c\left(o_{jp}\right) \leftarrow \hat{Q}_p^c + T_p\left(o_{jp}\right) + T_{cm}$
select $c^*$ such that $\arg\min_c \hat{f}_{cf}^c\left(o_{jp}\right)$
```

In the above procedure, μ_1 is a parameter to give an advantage to the current cell.

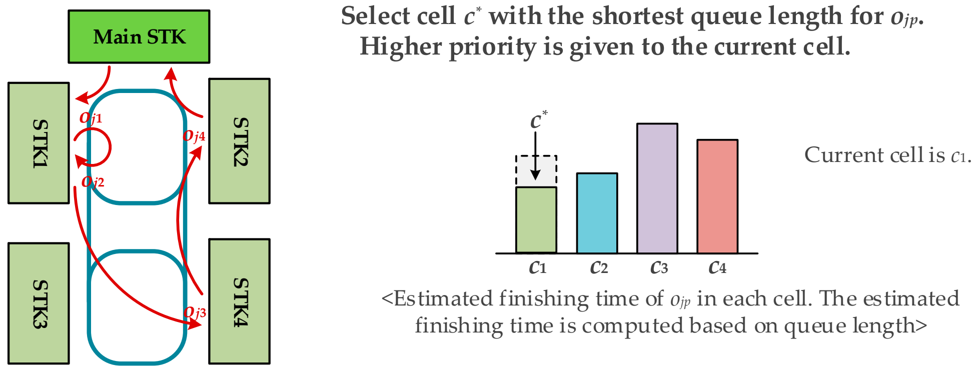

Main STK
STK1
STK2
STK3
STK4
o_{j1}
o_{j2}
o_{j3}
o_{j4}
Select cell c^* with the shortest queue length for o_{jp}.
Higher priority is given to the current cell.
c^*
Current cell is c_1.
c_1
c_2
c_3
c_4
<Estimated finishing time of o_{jp} in each cell. The estimated finishing time is computed based on queue length>

Figure 5. M-SSQL policy. In selection of a target cell for o_{jp}, similar to SSQL, it considers the queue length for o_{jp}. However, it gives an advantage to the current cell to reduce transfer time between cells.

4.1.4. Maximization of Direct Transfer

The BRD, SSQL, and M-SSQL are used to select a target cell only for a single subsequent operation of the job to be dispatched. However, the maximization of direct transfer (MDT) is used to select a target cell for the job to be dispatched as well as its proper partition set of remaining operations of the job to be processed in the designated target cell. Unlike the BRD, SSQL, and M-SSQL, the MDT permits machine-to-machine direct transfers in a cell, as shown in Figure 6. A machine-to-machine direct transfer in a cell can reduce the mean cycle time and utilities of intercell and intracell material handling systems. Let us assume that the sequence of remaining operations of job j is denoted by O_j^r. Then the number of possible partitions of remaining operation sequence is $|O_j^r|$. For instance, if $O_j^r = < o_{j3}, o_{j4}, o_{j5} >$, then the whole possible partitions, which should include consecutive operations starting from the first operation of O_j^r, are $< o_{j3} >$, $< o_{j3}, o_{j4} >$, and $< o_{j3}, o_{j4}, o_{j5} >$. Since the MDT considers machine-to-machine direct transfer, it should determine a proper partition of the remaining operation sequence, p_{jp}, and then decides its target cell $\chi(p_{jp})$. The partition of the remaining operation sequence, p_{jp}, and its target cell c^* is determined by the following procedure.

```
for each $c \in C$
  for each possible $p_{jp}$
    compute $\hat{q}(c, p_{jp})$
select $c^*, p_{jp}^*$ such that $\arg\min_{c, p_{jp}} \hat{q}(c, p_{jp})$
```

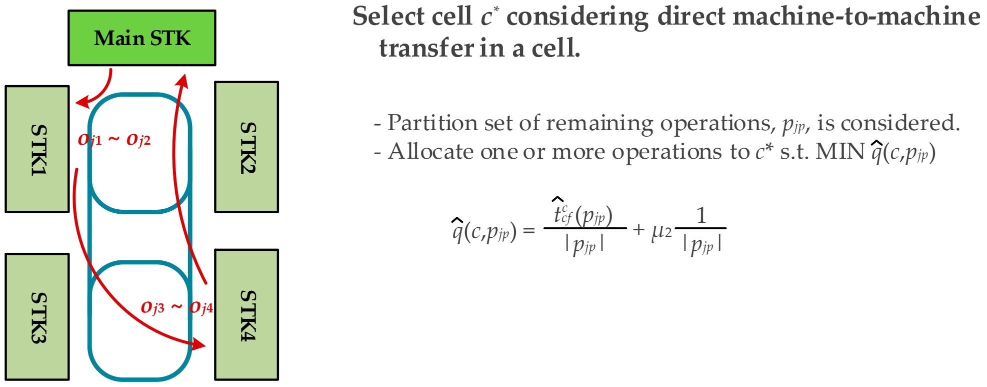

Main STK
STK1
STK2
STK3
STK4
$o_{j1} \sim o_{j2}$
$o_{j3} \sim o_{j4}$
Select cell c^* considering direct machine-to-machine transfer in a cell.
- Partition set of remaining operations, p_{jp}, is considered.
- Allocate one or more operations to c^* s.t. MIN $\hat{q}(c,p_{jp})$
$\hat{q}(c,p_{jp}) = \frac{\hat{t}_{cf}^{c}(p_{jp})}{|p_{jp}|} + \mu_2 \frac{1}{|p_{jp}|}$

Figure 6. Maximization of direct transfer (MDT) policy. It selects a target cell for a proper partition set of remaining operations, p_{jp}. If more than one of the operations are allocated to a target cell, they may be transferred machine-to-machine directly. This reduces the transfer time in a cell.

In the above procedure, $\hat{q}(c, p_{jp})$ is defined as follows.

$$\hat{q}(c, p_{jp}) = \frac{\hat{t}^{c}_{cf}(p_{jp})}{|p_{jp}|} + \mu_2 \frac{1}{|p_{jp}|}, \tag{4}$$

where μ_2 is a weight parameter.

4.2. Scheduling Policies for Ntra-Cell Scheduler

Two heuristic scheduling policies are investigated for the intracell scheduler. The first heuristic scheduling policy is designed for the case without machine-to-machine direct transfer in a cell, whereas the other policy is designed considering machine-to-machine direct transfer in a cell. The intracell scheduler selects a proper machine to be processed in each cell by determining the machine allocation function $\psi(o_{jp})$.

4.2.1. Shortest EQ Queue Length

The shortest EQ queue length (SEQL) is used to select a target machine for the next operation of the job to be dispatched in a cell, and is compatible with BRD, SSQL, and M-SSQL in terms of intercell policies. The SEQL selects the machine with the shortest input buffer's queue length among the machines whose job types are consistent with the next operation of the job to be dispatched. Under the SEQL policy, the job should be returned after completing its operation at the determined machine. The SEQL, which does not consider any machine-to-machine direct transfer in each cell, selects a job among the list of jobs waiting to be transferred and determines the target of the selected job in each cell as shown in Figure 7. Depending on the status of the job, its transfer type will be one of cell stocker-to-machine or machine-to-cell stocker. If the transfer type is cell stocker-to-machine, the target of the job is machine, otherwise, the target is cell stocker. The procedure of the SEQL rule is as follows.

```
select job $j^*$ such that $\arg\max_j [t_w(j) + \omega_t(j)]$
if the transfer type of $j^*$ is cell stocker-to-machine
    select the target machine $m^*$
    order $j^*$ to machine $m^*$
else
    else $j^*$ to the cell stocker
```

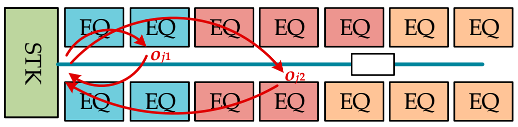


Figure 7. SEQL policy. It determines a target machine for an operation. The target machine with the shortest EQ queue length is selected. A job should be returned to stoker after completing its operation in the machine.

Line 1 select the job to be transferred by considering its waiting time $t_w(j)$ and weight $\omega_t(j)$ that is predefined according to the transfer type, i.e., cell stocker-to-machine and machine-to-cell stocker. The job whose transfer type is cell stocker-to-machine, has the higher priority than the job whose transfer type is machine-to-cell stocker. If the transfer type of the selected job is cell stocker-to-machine, in lines 3–4, the SEQL determines the target machine m^* and makes an order to transfer the job to m^*. If the transfer type of the selected job is machine-to-cell stocker, the SEQL makes an order to transfer the job to the cell stocker. Regarding the determination of a target machine for cell stocker-to-machine transfer, for a given operation o_{jp}, it selects the machine with the shortest EQ queue length. Each machine contains one or more input buffer(s) and one or more output buffer(s), and the EQ queue length is defined as the sum of the remaining processing time in a machine and the number of jobs in input buffer(s) multiplied by their processing times. The target machine m^* for o_{jp} , whose current cell is $c \in C$, is determined by the following procedure. The EQ queue length of machine m in cell c is denoted by QL_m^c.

```
$M_p^c \leftarrow \left\{ m \middle| m \in M^c \text{ and } \eta(m) = \delta\left(o_{jp}\right) \right\}$
if $M_p^c \neq \varnothing$
for each $m$ such that $m \in M_p^c$
    compute EQ queue length $QL_m^c$
select $m^*$ such that $\arg\min_m QL_m^c$
else
return NULL
```

4.2.2. Direct Transfer

The direct transfer (DT) is used to select a target machine for the next partition set of operations of the job to be dispatched in a cell, and it is compatible with MDT in terms of intercell policies. The partition set of operations can have one or more consecutive operations. Under the DT policy, the job should be returned after completing the entirety of its operations included in the partition set of operations. In determining a target machine, as with the SEQL, the DT also selects the machine with the shortest input buffer's queue length. With consideration of machine-to-machine direct transfer, DT selects a job among the list of jobs waiting to be transferred and determines the target of the job in each cell as shown in Figure 8. The transfer type of a job is one of machine-to-machine, cell stocker-to-machine, or machine-to-cell stocker. The procedure of the DT rule is as follows.

```
select job $j^*$ such that $\arg\max_j [t_w(j) + \omega_t(j)]$
if the transfer type of $j^*$ is machine-to-machine
    select the target machine $m^*$
    order $j^*$ to machine $m^*$
else if the transfer type of $j$ is cell stocker-to-machine
    select the target machine $m^*$
    order $j^*$ to machine $m^*$
else
    order $j^*$ to the cell stocker
```

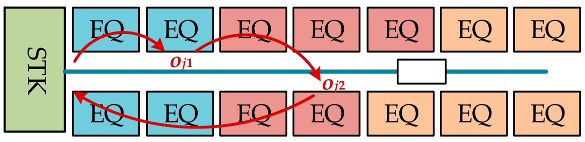


Figure 8. DT policy. It determines a target machine for an operation. If the next operation is to be processed in the same cell, it finds its next target machine.

The procedure of the DT rule is the same with that of the SEQL rule except that it considers the machine-to-machine transfer type. In line 1, when selecting a job to be transferred, the job with the transfer type of machine-to-machine has the highest priority over the other transfer types. Then, the transfer type of cell stocker-to-machine has the higher priority than that of machine-to-cell stocker. The procedure to determine the target machine m^* is same with that of the SEQL.

5. Simulation Experiments

The performances of the proposed heuristic scheduling policies are evaluated through computer simulation experiments, which re-executed using AutoMod Version 14.0.1 of Applied Materials, Inc. For the simulation experiments, the layout of a wafer test facility and operation flows are redesigned based on a real existing memory wafer test facility. It is assumed that the wafer test facility has six cells and that there is one main stocker and one cell stocker for every cell. Every cell contains 40 processing machines. A FOUP is used to transfer silicon wafers securely and safely for intercell transport and intracell transport. Each FOUP is assumed to have two wafers. Every machine has two input-buffers and two output-buffers to store FOUPs temporally. Table 1 shows the machine type and the processing time for each operation step of three types of wafers, and Table 2 shows the number of machines equipped in each cell. In the wafer test facility, machines are equipped functionally or multifunctionally in each cell, which is known as a hybrid cellular production line. Machines' configurations of cells 1–4 are designed so that a FOUP can execute its whole operation in a single cell, while those of cells 5–6 are designed to balance workloads among machine types. When wafer types and their processing times are fixed in advance, it would be the best machines' configuration if every cell contains a balanced number of machines for all operations. However, when a new type of wafer is released into the wafer test facility, the workloads' balance among machine types is broken. Hence, a lot of wafer test facilities adopt hybrid cellular production lines. Although the configuration of machines in each cell significantly affects the manufacturing performance indices such as work in process (WIP), cycle time, utilities of material handling systems, and optimal layout; machine configuration is not within the scope of this study. Thus, the proposed scheduling policies have been tested with the predefined layout and machines' configuration. Figure 9 shows a screenshot of Automod's simulation experiment.

Table 1. Processing times per wafer for three types of front open unified pods (FOUPs).

Operation	**Machine Type**	**Processing Times Per Wafer (seconds)**		
		Wafer A	**Wafer B**	**Wafer C**
OP1	MT1	1800	1200	1500
OP2	MT2	1080	1280	900
OP3	MT3	420	480	450
OP4	MT4	660	700	630
OP5	MT5	540	580	550

Table 2. Number of machines in each cell.

Cell	**MT1**	**MT2**	**MT3**	**MT4**	**MT5**
Cell 1	14	11	4	6	5
Cell 2	14	11	4	6	5
Cell 3	14	11	4	6	5
Cell 4	14	11	4	6	5
Cell 5	27	-	-	13	-
Cell 6	-	19	10	-	11

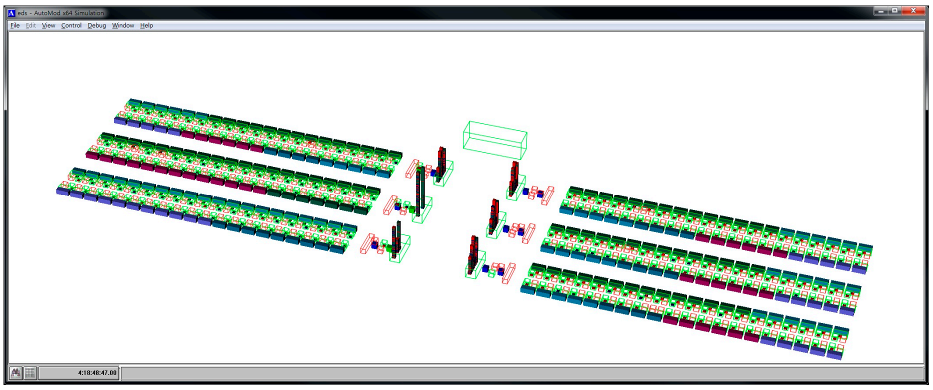

eds - AutoMod x64 Simulation
File Edit View Control Debug Window Help
4:18:48:47.00

Figure 9. AutoMod simulation.

For the simulation experiments, four combinations of intercell and intracell scheduling policies were investigated: BRD+SEQL (or shortly BRD), SSQL+SEQL (or shortly SSQL), M-SSQL+SEQL (or shortly M-SSQL), and MDT+DT (or shortly MDT). The tested combinations of scheduling policies can be classified into two scenario groups. One is the combinations of scheduling policies without consideration given to the machine-to-machine direct transfer in each cell, and the other is those with consideration given to the machine-to-machine direct transfers. For the former group, the BRD, SSQL, and M-SSQL policies are investigated for the intercell scheduling policies. In this scenario group, SEQL is applied for the intracell scheduling policies. On the other hand, for the latter group, the MDT and DT policies are developed and investigated for intercell and intracell scheduling, respectively. The MDT policy is designed to only be compatible with the DT policy. The release of FOUPs is controlled so that the average throughputs of the four scheduling policies meets the near-maximum capacity of the wafer test facility.

One of the most important indices in manufacturing systems is throughput. Figure 10 depicts the daily throughput trends of the four scheduling policies for ten days. The average throughputs are 2379.8, 2400.0, 2398.6, and 2388.3 FOUPs/day under BRD, SSQL, M-SSQL, and MDT, respectively. Although there are no significant differences among the average throughputs under the four scheduling policies, as shown in the graph, the daily perturbation of throughputs is the lowest under the MDT policy.

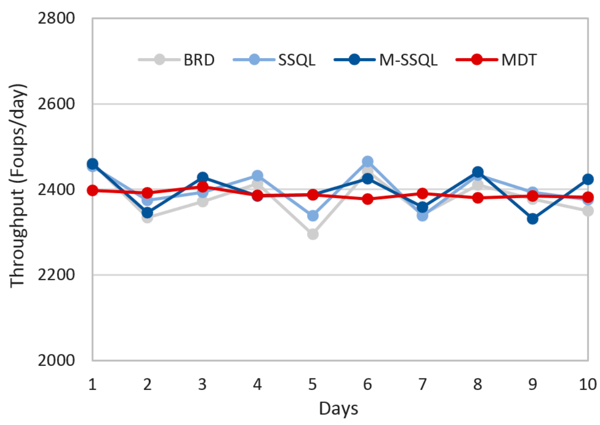


Figure 10. Daily throughputs of the four scheduling policies.

The WIP is another important production index. Figure 11 shows the trends of the WIP under the four scheduling policies. The average WIPs are 2410.1, 2307.1, 2369.7, and 1997.3 FOUPs under BRD, SSQL, M-SSQL, and MDT, respectively. There are no significant differences among the average WIPs under BRD, SSQL, and M-SSQL, whereas that under MDT shows the lowest WIP level. In addition, the daily WIP variation is the lowest under the MDT policy. More specifically, the daily average number of FOUPs in each stocker are investigated in Figure 12. In cells 1 to 3, where the machine configurations are designed so that a FOUP can execute all of its operations in a single cell, there are no significant differences among the average number of FOUPs under the four scheduling policies. However, the MDT shows the lowest daily variation of FOUP numbers. In cells 4 to 6, on the other hand, there are differences among the average number of FOUPs, but the MDT also shows the lowest daily variation of FOUP numbers.

With respect to the daily cycle time, as shown in Figure 13, the MDT scheduling policy shows the lowest value. For instance, the average cycle times of Wafer Type A under the BRD, SSQL, M-SSQL, and MDT rules are 24.5, 23.7, 21.1, and 20.3 days, respectively, and those of Wafer Type B are 23.7, 22.6, 20.6, and 20.1 days, respectively. The average cycle times for Wafer Type C are 25.0, 24.2, 21.1, and 20.2 days, respectively. The MDT, M-SSQL, SSQL, and BRD rules show lower to higher average cycle times in the listed order. The MDT scheduling policy also shows the lowest variations of daily cycle times among the four scheduling policies. Cycle time variations have recently been considered to be a significant production index in the semiconductor industry because lower daily cycle time variations are helpful in terms of order due dates.

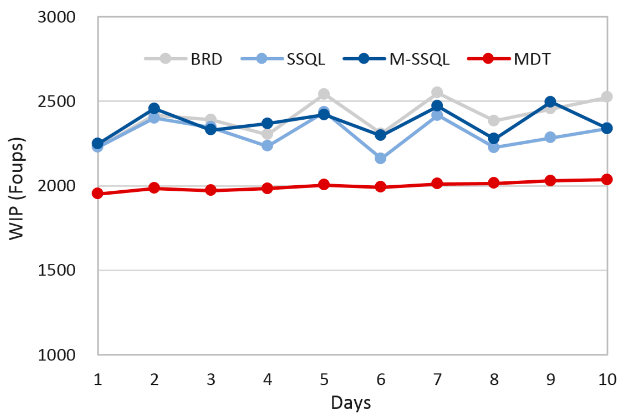


Figure 11. Daily work in process of the four scheduling policies.

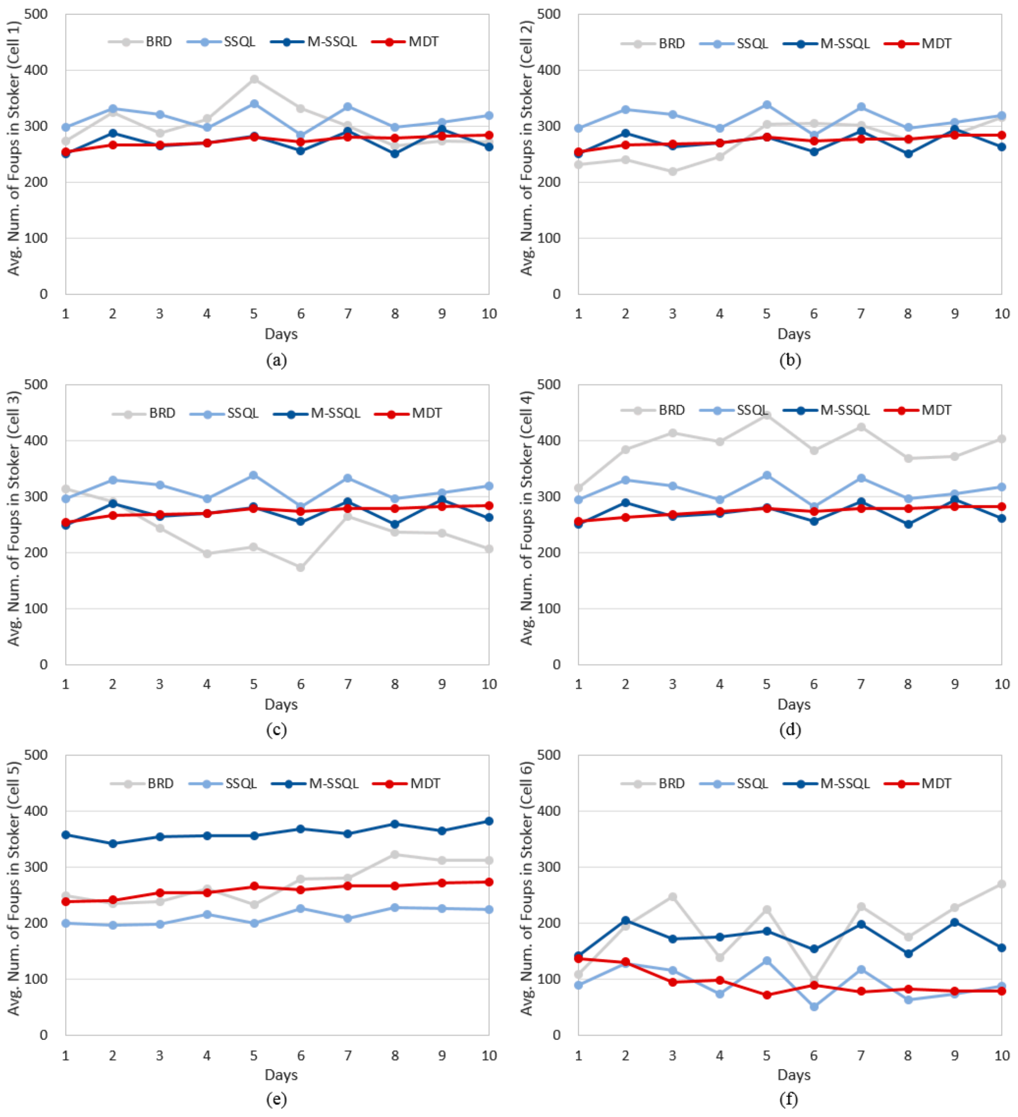


Figure 12. Daily average number of FOUPs in stocker of (**a**) cell 1, (**b**) cell 2, (**c**) cell 3, (**d**) cell 4, (**e**) cell 5, and (**f**) cell 6.

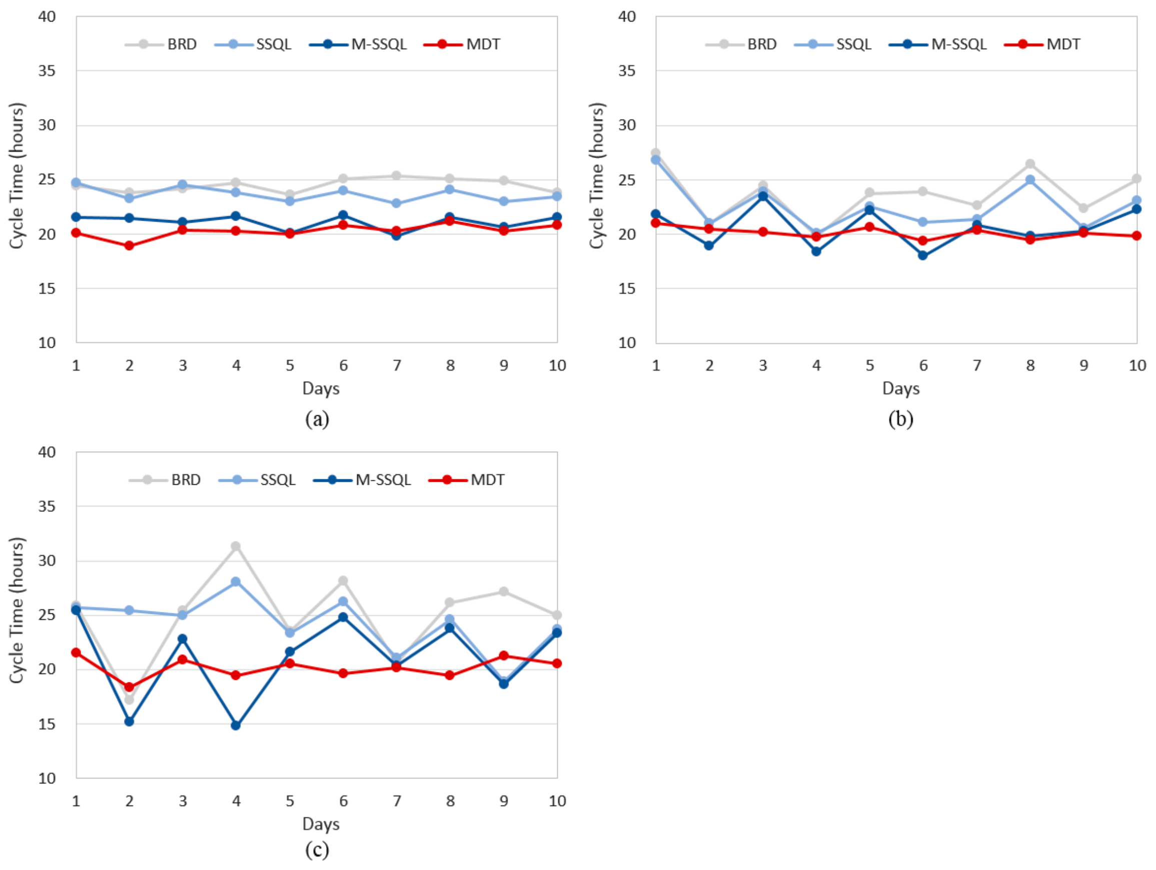


Figure 13. Daily cycle times of the four scheduling policies of (**a**) wafer type A, (**b**) wafer type B, and (**c**) wafer type C.

With respect to intercell and intracell material handling systems, graphs showing the daily average number of claims are shown in Figure 14; Figure 15, respectively. The daily average number of claims are investigated instead of the utilities of material handling vehicles because the number of material handling vehicles is determined by considering the number of claims in the general semiconductor industry. Lower job claim numbers can reduce the number of material handling vehicles, and fewer variations in job claim numbers can also reduce the number of material handling vehicles. Figure 14 shows the daily average number of claims on OHT (Overhead Transport), or the intercell material handling system. As anticipated, the MDT scheduling policy significantly reduces the daily average number of claims compared with the BRD, SSQL, and M-SSQL policies. This is because the transfers from cell to cell are reduced by applying the MDT policy. The daily average numbers of claims on OHT are 24.8, 24.9, 24.9, and 14.0 under the BRD, SSQL, M-SSQL, and MDT, respectively. Figure 15 shows the daily average number of claims on AGV (Automated Guided Vehicle), or the intracell material handling system, in each cell. In cells 1 to 4, where the machine configurations are designed so that a FOUP can execute all of its operations in a single cell, the MDT policy reduces the number of claims by 27.4 to 27.5%. The machine types equipped in cell 5 are MT1 and MT4, and therefore, there is no machine-to-machine direct transfer via the MDT policy. Thus, the average numbers of claims in cell 5 have no significant differences among the four scheduling policies. On the other hand, the machine types equipped in cell 6 are MT2, MT3, and MT5, and therefore, machine-to-machine transfers from MT2 to MT3 are possible under the MDT policies. Thus, the average number of claims under the MDT is lower than those under the other three policies.

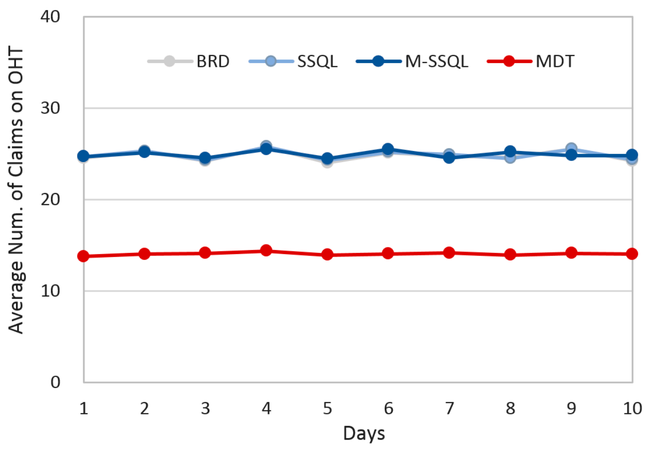


Figure 14. Daily average number of claims on OHT.

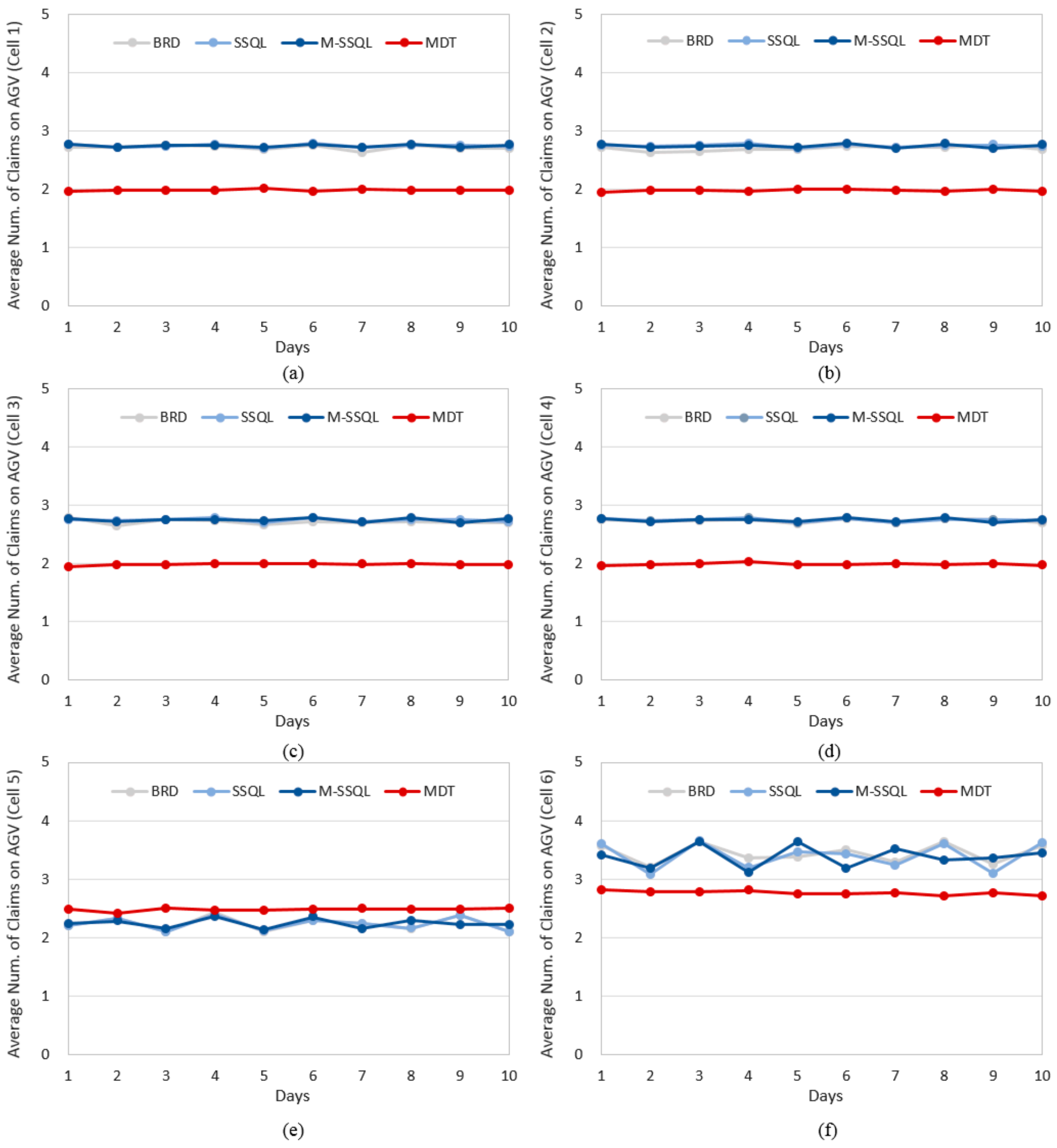


Figure 15. Daily average number of claims on AGV in (**a**) cell 1, (**b**) cell 2, (**c**) cell 3, (**d**) cell 4, (**e**) cell 5, and (**f**) cell 6.

6. Conclusions

This paper explores the effects of several scheduling policies on an EDS test facility. The simulation experiment for each combination of intercell and intracell scheduling policies are presented for a given EDS facility. Wafer fabrication facilities, which are generally considered to be complex job shops, have standardized layouts and structures in terms of the material handling system. Unlike wafer fabrication facilities, however, wafer test facilities actually have a variety of layouts and hierarchies for their material handling systems. Many semiconductor vendors have different equipment layouts and have different logistics systems, and therefore, their scheduling problem definitions and methodologies should differ accordingly. The configuration of the layout including cell, stocker, and test machines are based on real industries and adopt data as much as possible for experimental reliability. We conclude that the MDT, which maximizes the direct travel from machine to machine by minimizing the stockers' interference, is superior to the other policy combinations for its cycle time as well as WIP.

The given configuration is for a certain semiconductor memory product, and the layout as well as machine allocation, could be changed to improve efficiency by reducing the effort of material handling. Thus, a consideration of the robustness of the operation rule is important for long term semiconductor facility planning and operation. As such, future work could focus on further explorations into the adoptive policy for different products.

Author Contributions: Conceptualization, H.J.Y and J.C.; Methodology, H.J.Y. and J.C.; Software, H.J.Y.; Validation, H.J.Y and J.C.; Formal Analysis, H.J.Y.; Investigation, H.J.Y and J.C.; Resources, H.J.Y and J.C.; Data Curation, H.J.Y. and J.C.; Writing—Original Draft Preparation, H.J.Y and J.C.; Writing—Review and Editing, H.J.Y. and J.C.; Visualization, H.J.Y. and J.C.

Funding: For this work, Hyun Joong Yoon was supported by research grants from Daegu Catholic University in 2018.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Sarkis, J. Manufacturing's role in corporate environmental sustainability—Concerns for the new millennium. *Int. J. Oper. Prod. Manag.* **2001**, *21*, 666–686. [CrossRef]
2. Rosen, M.A.; Kishawy, H.A. Sustainable manufacturing and design: Concepts, practices and needs. *Sustainability* **2012**, *4*, 154–174. [CrossRef]
3. Jayal, A.D.; Badurdeen, F.; Dillon, O.W., Jr.; Jawahir, I.S. Sustainable manufacturing: Modeling and optimization challenges at the product, process and system levels. *CIRP J. Manuf. Sci. Technol.* **2010**, *2*, 144–152. [CrossRef]
4. Bang, J.Y.; Kim, Y.D. Scheduling algorithms for a semiconductor probing facility. *Comput. Oper. Res.* **2011**, *38*, 666–673. [CrossRef]
5. Lin, J.T.; Wang, F.K.; Kuo, P.C. A parameterized-dispatching rule for a Logic IC sort in a wafer fabrication. *Prod. Plan. Control Manag. Oper.* **2005**, *16*, 426–436. [CrossRef]
6. Uzsoy, R.; Martin-Vega, L.A.; Lee, C.Y.; Leonard, P.A. Production scheduling algorithms for a semiconductor test facility. *Semicond. Manuf. IEEE Trans.* **1991**, *4*, 270–280. [CrossRef]
7. Ovacik, I.M.; Uzsoy, R. A shifting bottleneck algorithm for scheduling semiconductor testing operations. *J. Electron. Manuf.* **1992**, *2*. [CrossRef]
8. Ovacik, I.M.; Uzsoy, R. Decomposition methods for scheduling semiconductor testing facilities. *Int. J. Flex. Manuf. Syst.* **1996**, *8*, 357–387. [CrossRef]
9. Chen, T.-R.; Chang, T.-S.; Chen, C.-W.; Kao, J. Scheduling for IC sort and test with preemptiveness via Lagrangian relaxation. *IEEE Trans. Syst. Man. Cybern.* **1995**, *25*, 1249–1256. [CrossRef]
10. Chen, T.-R.; Hsia, T.C. Scheduling for IC sort and test facilities with precedence constraints via lagrangian relaxation. *J. Manuf. Syst.* **1997**, *16*, 117–128. [CrossRef]
11. Huang, S.-C.; Lin, J.T. An interactive scheduler for a wafer probe centre in semiconductor manufacturing. *Int. J. Prod. Res.* **1998**, *36*, 1883–1900. [CrossRef]

12. De, S.; Lee, A. Towards a knowledge-based scheduling system for semiconductor testing. *Int. J. Prod. Res.* **1998**, *36*, 1045–1073. [CrossRef]
13. Yang, J.; Chang, T.-S. Multiobjective scheduling for IC sort and test with a simulation testbed. *IEEE Trans. Semicond. Manuf.* **1998**, *11*, 304–315. [CrossRef]
14. Xiong, H.H.; Zhou, M. Scheduling of semiconductor test facility via Petri nets and hybrid heuristic search. *IEEE Trans. Semicond. Manuf.* **1998**, *11*, 384–393. [CrossRef]
15. Sivakumar, A.I. Optimization of cycle time and utilization in semiconductor test manufacturing uinsg simulation based, on-line, near-real-time scheduling system. In Proceedings of the 1999 Winter Simulation Conference, Phonix, AZ, USA, 5–8 December 1999; pp. 727–735.
16. Lee, Y.H.; Lee, B.K.; Jeong, B. Multi-objective production scheduling of probe process in semiconductor manufacturing. *Prod. Plan. Control Manag. Oper.* **2000**, *11*, 660–669. [CrossRef]
17. Pearn, W.L.; Chung, S.H.; Yang, M.H. A case study on the wafer probing scheduling problem. *Prod. Plan. Control Manag. Oper.* **2002**, *13*, 66–75. [CrossRef]
18. Pearn, W.L.; Chung, S.H.; Yang, M.H. Minimizing the total machine workload for the wafer probing scheduling problem. *IIE Trans. Institute Ind. Eng.* **2002**, *34*, 211–220. [CrossRef]
19. Pearn, W.L.; Chung, S.H.; Yang, M.H.; Chen, Y.H. Algorithms for the wafer probing scheduling problem with sequence-dependent set-up time and due date restrictions. *J. Oper. Res. Soc.* **2004**, *55*, 1194–1207. [CrossRef]
20. Pearn, W.L.; Chung, S.H.; Chen, A.Y.; Yang, M.H. A case study on the multistage IC final testing scheduling problem with reentry. *Int. J. Prod. Econ.* **2004**, *88*, 257–267. [CrossRef]
21. Chiang, T.; Shen, Y.; Fu, L. Adaptive Lot/Equipment Matching Strategy and GA Based Approach for Optimized Dispatching and Scheduling in a Wafer Probe Center. In Proceedings of the 2004 IEEE International Conference on Robotics & Automation, New Orleans, LA, USA, 26 April–1 May 2004; pp. 3125–3130.
22. Chiang, T.C.; Shen, Y.S.; Fu, L.C. A new paradigm for rule-based scheduling in the wafer probe centre. *Int. J. Prod. Res.* **2008**, *46*, 4111–4133. [CrossRef]
23. Lin, J.T.; Wang, F.K.; Lee, W.T. Capacity-constrained scheduling for a logic IC final test facility. *Int. J. Prod. Res.* **2004**, *42*, 79–99. [CrossRef]
24. Song, Y.; Zhang, M.T.; Yi, J.; Zhang, L.; Zheng, L. Bottleneck station scheduling in semiconductor assembly and test manufacturing using ant colony optimization. *IEEE Trans. Autom. Sci. Eng.* **2007**, *4*, 569–578. [CrossRef]
25. Pearn, W.L.; Chung, S.H.; Shiao, K.P. Solution strategies for multi-stage wafer probing scheduling problem with reentry. *J. Oper. Res. Soc.* **2008**, *59*, 637–651. [CrossRef]
26. Lee, Y.H.; Ham, M.; Yoo, B.; Lee, J.S. Daily planning and scheduling system for the EDS process in a semiconductor manufacturing facility. *Int. J. Adv. Manuf. Technol.* **2009**, *41*, 568–579. [CrossRef]
27. Lin, S.-W.; Lee, Z.-J.; Ying, K.-C.; Lin, R.-H. Meta-heuristic algorithms for wafer sorting scheduling problems. *J. Oper. Res. Soc.* **2011**, *62*, 165–174. [CrossRef]
28. Ying, K.-C. Scheduling identical wafer sorting parallel machines with sequence-dependent setup times using an iterated greedy heuristic. *Int. J. Prod. Res.* **2012**, *50*, 2710–2719. [CrossRef]
29. Jacobs, L.W.; Brusco, M.J. A local-search heuristic for large set-covering problems. *Nav. Res. Logist. Q.* **1995**, *42*, 1129–1140. [CrossRef]
30. Doleschal, D.; Lange, J.; Weigert, G.; Klemmt, A. Improving flow line scheduling by upstream mixed integer resource allocation in a wafer test facility. In Proceedings of the 2012 Winter Simulation Conference, Berlin, Germany, 9–12 December 2012.
31. Bard, J.F.; Gao, Z.; Chacon, R.; Stuber, J. Daily scheduling of multi-pass lots at assembly and test facilities. *Int. J. Prod. Res.* **2013**, *51*, 7047–7070. [CrossRef]
32. Pearn, W.L.; Hong, J.S.; Tai, Y.T. The burn-in test scheduling problem with batch dependent processing time and sequence dependent setup time. *Int. J. Prod. Res.* **2013**, *51*, 1694–1706. [CrossRef]
33. Lee, C.Y.; Uzsoy, R.; Martin-Vega, L.A. Efficient algorithms for scheduling semiconductor burn-in operations. *Oper. Res.* **1992**, *40*, 764–775. [CrossRef]
34. Hao, X.C.; Wu, J.Z.; Chien, C.F.; Gen, M. The cooperative estimation of distribution algorithm: A novel approach for semiconductor final test scheduling problems. *J. Intell. Manuf.* **2014**, *25*, 867–879. [CrossRef]
35. Zheng, X.L.; Wang, L.; Wang, S.Y. A novel fruit fly optimization algorithm for the semiconductor final testing scheduling problem. *Knowledge-Based Syst.* **2014**, *57*, 95–103. [CrossRef]

36. Bard, J.F.; Jia, S.; Chacon, R.; Stuber, J. Integrating optimisation and simulation approaches for daily scheduling of assembly and test operations. *Int. J. Prod. Res.* **2015**, *53*, 2617–2632. [CrossRef]
37. Wang, S.; Wang, L.; Liu, M.; Xu, Y. A hybrid estimation of distribution algorithm for the semiconductor final testing scheduling problem. *J. Intell. Manuf.* **2015**, *26*, 861–871. [CrossRef]
38. Kim, J.; Chung, S.Y.; Yoon, H.J. Multi-agent-based scheduling methods for hybrid cellular production lines in semiconductor industry. *Proc. Inst. Mech. Eng. Part B J. Eng. Manuf.* **2014**, *228*, 1701–1712. [CrossRef]
39. Gao, Z.; Bard, J.F.; Chacon, R.; Stuber, J. IIE Transactions An assignment-sequencing methodology for scheduling assembly and test operations with multi-pass requirements An assignment-sequencing methodology for scheduling assembly and test operations with multi-pass requirements. *IIE Trans.* **2015**, *47*, 153–172. [CrossRef]
40. Jia, S.; Bard, J.F.; Chacon, R.; Stuber, J. Improving performance of dispatch rules for daily scheduling of assembly and test operations. *Comput. Ind. Eng.* **2015**, *90*, 86–106. [CrossRef]
41. Hur, Y.; Bard, J.F.; Chacon, R. Hierarchy machine set-up for multi-pass lot scheduling at semiconductor assembly and test facilities. *Int. J. Prod. Res.* **2017**. [CrossRef]
42. Jia, S.; Morrice, D.J.; Bard, J.F. A performance analysis of dispatch rules for semiconductor assembly & test operations. *J. Simul.* **2018**. [CrossRef]
43. Wang, S.; Wang, L. A knowledge-based multi-agent evolutionary algorithm for semiconductor final testing scheduling problem. *Knowledge-Based Syst.* **2015**, *84*, 1–9. [CrossRef]
44. 4Joung, Y.M.; He, T.; Yoon, S.W.; Vancheeswaran, R.; Abela, C.; Andres, H.R. Multi-pass Lot Scheduling Algorithm for Maximizing Throughput at Semiconductor Final Test Facilities. *Procedia Manuf.* **2017**, *11*, 1992–2000.
45. Sang, H.-Y.; Duan, P.-Y.; Li, J.-Q. An effective invasive weed optimization algorithm for scheduling semiconductor final testing problem. *Swarm Evol. Comput.* **2018**, *38*, 42–53. [CrossRef]