Litho Machine Scheduling With Convex Hull Analyses

Bing Yan, *Student Member, IEEE*, Hsin Yuan Chen, Peter B. Luh, *Fellow, IEEE*, Simon Wang, and Joey Chang

***Abstract*—The increasing pressure to meet demand are forcing semiconductor manufacturers to seek efficient scheduling methods. Lithography, with a limited number of expensive resources and the reentrant nature of the fabrication processes, is a major bottleneck. This paper presents a litho machine scheduling formulation for high-volume and low-variety manufacturing over a day, with novel modeling of resource setups, reticle expirations, and future stacking layer load balancing. The problem is believed to be NP hard. After linearization and simplification, it is solved by using the branch-and-cut method by exploiting problem linearity. Near-optimal solutions for practical problems, however, are still difficult to obtain efficiently. Through detailed analyses, it was discovered that the convex hull of the problem is difficult to delineate and many low-efficient branching operations are needed. A two-phase approach is therefore established. In the first phase, a simplified problem with certain complicating constraints dropped is efficiently solved by exploiting linearity to reduce ranges of decision variables. The problem with the full set of constraints is then solved in the second phase with a much reduced decision space. Numerical testing shows that this two-phase approach can generate near-optimal schedules within reasonable amounts of computation time. This two-phase approach is generic, and will have major implications on other semiconductor scheduling problems and beyond.**

***Note to Practitioners*—Lithography is a major bottleneck in semiconductor manufacturing. This paper addresses the scheduling of litho machines and reticles over a day for high-volume and low-variety lots to meet daily targets of different products. A scheduling formulation with novel modeling of resource setups, reticle expirations, and future stacking layer load balancing is presented. The problem is solved by using a standard commercial solver that exploits problem linearity. Near-optimal schedules, however, are difficult to obtain efficiently for practical problems. Through detailed analyses, it was discovered that the difficulty is caused by certain constraints in the formulation. A two-phase approach is therefore developed. In the first phase, a simplified problem with complicating constraints dropped is efficiently solved to reduce ranges of decision variables. The problem with the full set of constraints is then solved in the second phase with a much reduced decision space. This method can obtain near-optimal schedules for practical litho machine scheduling problems within reasonable amounts of computation time.**

***Index Terms*—Branch-and-cut, convex hull, litho machine scheduling, mixed-integer optimization, semiconductor manufacturing, two-phase approach.**

I. Introduction

LITHOGRAPHY is the process of transferring circuit patterns to the surface of a wafer by selectively exposing light through a reticle. During this process, a wafer is incrementally developed layer by layer in lots [16], where different products require different sets of layers to be completed. Lithography, with a limited number of expensive resources and the reentrant nature of the fabrication processes, is a major bottleneck in semiconductor manufacturing [1]. The increasing pressure to meet demand is forcing manufacturers to seek efficient scheduling methods.

In a fab, litho machines are generally unique, and reticles are usually divided into groups based on which product\layer they process. One machine usually requires one reticle to process a layer. Before processing a specific layer, a machine and a corresponding reticle need to be set up, and excessive setups are costly and undesirable. During processing, a lot needs a certain amount of time to be completed. In addition, reticles need to be recalibrated after processing a certain number of lots. Reticles in the same group therefore should not expire simultaneously to avoid reticle shortage. For certain products, a selected set of layers (stacking layers) must be processed on the same machine for precision fabrication. The load on machines processing stacking layers need to be balanced to prevent future overload or starvation. In our problem, products have high volume and low variety, and a daily target is assigned to each product/layer. Therefore, there is no need to number and distinguish each lot. The problem is to allocate machines and reticles over a day to meet the daily targets.

As will be reviewed in Section II, reticle expiration was rarely addressed in the literature. Also, most papers focused on balancing the current load, and rarely discussed the effect of machine assignments on future load through stacking layers. In this paper, a formulation for litho machine scheduling over a day is established with novel modeling of resource setups, reticle expirations, and future stacking layer load in Section III. It contains four major sets of constraints regarding resource capacities, processing times, maximal numbers of lots scheduled and setups. Since a setup time is generally much shorter than the corresponding lot processing time, the time of setup is ignored and the number of setups is considered. To simplify the formu-

Manuscript received July 01, 2013; accepted July 27, 2013. Date of publication August 22, 2013; date of current version October 02, 2013. This paper was recommended for publication by Associate Editor W. Shen and Editor M. C. Zhou upon evaluation of the reviewers' comments. This work was supported in part by Inotera Memories Inc. The views expressed in this paper are solely those of the authors and do not necessarily represent those of Inotera Memories Inc.

B. Yan and P. B. Luh are with the Department of Electrical and Computer Engineering, University of Connecticut, Storrs, CT 06269-4157 USA (e-mail: bing.yan@uconn.edu; Peter.Luh@uconn.edu).

H. Y. Chen, S. Wang, and J. Chang are with Inotera Memories Inc., Kueishan, Taoyuan 333, Taiwan (e-mail: gary1311@inotera.com; simonwang@inotera.com; chang_joey@yahoo.com).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TASE.2013.2277812

lation and to reduce the number of setups, it is assumed that all the lots assigned to a machine to process a particular layer within the day will be processed under one setup. The objective function is to meet targets, balance future load, avoid simultaneous reticle expirations, and avoid excessive setups. Future stacking layer load can be adjusted through proper machine assignments, and simultaneous reticle expirations can be avoided by spacing out expiration dates through proper reticle assignments. The problem formulated above is linear and believed to be NP hard.

The problem is solved by using the branch-and-cut method in Section IV after certain constraints are simplified without sacrificing optimality. Branch-and-cut is powerful for certain classes of mixed-integer linear optimization problems, and is easy to code by using commercial solvers. In the method, the integrality-relaxed problem is solved first by using a linear programming method. If all integer decision variables are integers, the solution is optimal to the original problem. If not, valid cuts are added trying to obtain the convex hull. The idea is that once the convex hull is obtained, all integer decision variables of the linear programming solution are integers and optimal to the original problem. The process of obtaining the convex hull, however, is problem dependent, and can itself be NP hard. Low-efficient branching operations may then be needed. We found that near-optimal solutions for practical problems are difficult to obtain efficiently. Through detailed analyses, it was discovered that the convex hull is difficult to delineate because of certain complicating constraints. A two-phase approach is therefore developed. In the first phase, a simplified problem with those complicating constraints dropped is efficiently solved to establish ranges of decision variables. The problem with the full set of constraints is then solved in the second phase with a much reduced decision space.

The methods have been implemented by using IBM ILOG CPLEX, and three examples are presented in Section V. Numerical results show that the two-phase approach can generate near-optimal schedules within much reduced computation time than the one-phase approach. More importantly, this approach is generic, and will have major implications on other semiconductor scheduling problems and beyond.

II. Literature Review

Developing effective scheduling approaches for semiconductor manufacturing is challenging because of its complex reentrant characteristics and the large sizes of practical problems. Approaches for litho machine scheduling including heuristic rules and mathematical programming will be reviewed.

A. Heuristic Rules With Simulation Techniques

Heuristic rules for litho machine scheduling are briefly reviewed with simulation techniques used to valid them in most papers. A mixed-integer model for short-time capacity scheduling was developed in [17], and the objective is to maximize throughput and the total amount of WIP processed at each workcenter including lithography. Fast heuristics were presented for computation efficiency. However, only a single product type was considered. A lot scheduling problem with capacity scheduling and lot sequencing subproblems was discussed in [2]. Greedy heuristics were used to solve the problem, and a simulation model of a wafer fabrication facility was used to examine the effects of this method on lithography. For simplicity, the processing time of each layer required by each lot was assumed to be identical on all machines. A method for load balancing in the lithography area based on the greedy algorithm was discussed in [14]. A detailed simulation model was developed. To improve load balancing in the lithography area, the lot assignment was decided at the time when the lots were released. Three dispatching rules and four bottleneck scheduling rules for lithography were studied in [12], and the objective is to maximize the production volume. Some lot scheduling rules were also developed for WIP balancing, and combinations of these rules were tested for various performance measurements. This study was extended in [3], where machines were eligible to process a specified subset of operations, and a setup was required when an operation was changed. The focus was on allocating the capacity to available jobs rather than making sequencing decisions. A number of heuristic algorithms and a greedy randomized adaptive search procedure were presented. A model that characterized the lithography process was developed in [4], and dispatching rules for mask change reduction and setup reduction were studied. Dispatching strategies for regular lots and priority lots were investigated in [20] to decrease cycle times and increase the number of daily moves. To balance load, the "Resource Schedule and Execution Matrix" model was presented in [16], and the lot with the largest wait steps was assigned to the litho machine with the smallest load. For simplicity, it was assumed that each lot had the same process steps and quantity, and each layer had the same processing time. With heuristic rules, schedules can be efficiently obtained, but it is difficult to find or know the optimal rules. Also, simulation can be time consuming.

B. Mathematical Programming

Mathematical programming methods including Lagrangian relaxation, branch-and-bound and branch-and-cut that have the capability to solve our problem are reviewed in this subsection. Lagrangian relaxation is a popular method for mathematical programming. A real-time scheduling and dispatching framework was developed in [9] for a semiconductor fab including lithography. The problem was solved by using Lagrangian relaxation and network flow techniques without considering setups. Lagrange relaxation was also used to solve a lot scheduling problem with aggregated process steps for high-variety and low-volume fabrication in [13]. Only problems with short planning horizons (e.g., one shift to one day) were considered due to complexity issues.

Branch-and-bound has also been used. A production control method was investigated in [18], and it was applied to discrete-event reentrant semiconductor manufacturing lines for scheduling. A tradeoff was made between production rate and cycle time for overall optimality. A mixed-integer stochastic programming model for capacity planning under demand uncertainty was developed in [7]. Cutting planes and a heuristic approach were used to improve computation efficiency of branch-and-bound. Still, computation efficiency remained to

be challenging for problems with larger numbers of scenarios and long periods.

Branch-and-cut has now been widely used. A WIP balancing concept was presented in [10], and the bottleneck machines were divided into different load levels for higher throughput. The mixed-integer formulation was solved by using CPLEX to decide the quantity of lots to be processed on litho machines. It was believed that the model with lot precedence constraints would require longer computation time. Branch-and-cut was also used to solve a single machine and multiple-lot-per-carrier (front-opening unified pod) scheduling problem in [15], and the objective was to minimize the sum of lot completion time. All carriers were assumed identical, and the processing time per wafer was assumed the same. The method could solve a problem at the root node itself, while it could not solve large-sized instances. It can be seen that for the papers with branch-and-cut, how to improve computation efficiency is a major challenge.

III. Problem Formulation

As reviewed in Section II, reticle expiration was rarely addressed in the literature. Also, most papers focused on balancing the current load, and rarely discussed the effect of machine assignments on future load through stacking layers. A novel formulation for litho machine scheduling over a day is established in this section. It contains four major sets of constraints as presented in the first four subsections. The objective function is to meet targets, balance future load, avoid simultaneous reticle expirations, and avoid excessive setups as discussed in Section III-E. To solve the problem by using branch-and-cut, a linear formulation is needed.

A. Resource Capacity Constraints

Consider a fab with M litho machines indexed by m and R reticles indexed by r as resources. There are K discrete time slots indexed by k within a day. In the fab, P types of products with index p are processed, and each requires L types of layers with index l.

For one machine or reticle, there are only three statuses, processing, idle, and unavailable. To obtain these statuses, a set of binary variables with machine, reticle and time indices is used here. Based on the formulation in [19], the key decision variables are defined as follows:

$$\delta_{mr}(k) = \begin{cases} 1, & \text{if machine } m \text{ is combined with reticle } r \\ & \text{to process a layer at time slot } k; \\ 0, & \text{otherwise.} \end{cases}$$

Machine capacity, reticle capacity, machine-reticle matching, and resource maintenance constraints are described as follows.

1) Machine Capacity Constraints: One machine requires only one reticle to process a layer at any time slot, i.e.,

$$\sum_{r} \delta_{mr}(k) \le 1, \quad \forall k, \forall m. \tag{1}$$

2) Reticle Capacity Constraints: Likewise, one reticle requires only one machine to process a layer at any time slot, i.e.,

$$\sum_{m} \delta_{mr}(k) \le 1, \quad \forall k, \forall r. \tag{2}$$

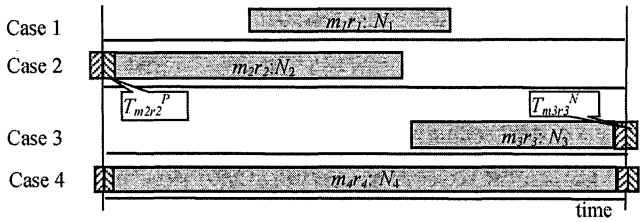


Fig. 1. Four situations of processing.

3) Machine-Reticle Matching Constraints: Litho machines are generally unique, and machine m cannot be combined with the reticles in set S_m^{RN} to process layers, i.e.,

$$\delta_{mr}(k) = 0, \quad \forall k, \forall m, \text{ and } r \in S_m^{RN}. \tag{3}$$

4) Resource Maintenance Constraints: One machine is not available during maintenance, i.e.,

$$\delta_{mr}(k) = 0, \quad k \in \left[b_m^M, c_m^M\right], \ m \in S^{MM} \text{ and } \forall r \tag{4}$$

where S^{MM} is the set of machines that need to do maintenance within the day, and b_m^M and c_m^M are the beginning time and the completion time of maintenance on machine m.

The modeling of reticle maintenance is similar

B. Processing Time Requirements

As mentioned earlier, the time of setup is ignored and the number of setups is considered since a setup time is generally much shorter than the corresponding lot processing time. To simplify the formulation and to reduce the number of setups, it is assumed that all the lots assigned to a machine to process a particular layer within the day will be processed under one setup. As shown in Fig. 1, machine m_1 with reticle r_1 processes N_1 lots under one setup in Case 1. Sometimes, there may be an unfinished lot on the machine at the beginning or end point of the day as shown in Cases 2 and 3, respectively. In Case 4, both of these two situations occur.

In general, let N_{mr} denote the number of lots scheduled on machine m and reticle r within the day, and N_{mr}^{UB} denote its upper bound. Let T_{mr}^P denote the time required to complete the unfinished lot left over from the previous day on machine m and reticle r, and T_{mr}^N denote the time required to complete the unfinished lot left for the next day. The value of the first variable is known, and the second variable is an integer decision variable. For machine m with reticle r, N_{mr} times of processing time T_{mr} must be assigned, and if the last time slot is involved, one unfinished lot can be left. The four cases mentioned above can be combined together as follows:

$$\sum_{k} \delta_{mr}(k) - T_{mr}^P + T_{mr}^N = N_{mr} \times T_{mr}, \tag{5}$$

$$0 \le N_{mr} \le N_{mr}^{UB}, N_{mr}^{UB} = \left(K - T_{mr}^P\right) \div T_{mr} + 1, \ \forall r, \forall m. \tag{6}$$

The unfinished lot from the previous day is assumed not to be included in the total number of lots scheduled within the day, while the lot left for the next day is.

In addition, the time required to process the unfinished lot left for the next day T_{mr}^{N} must be smaller than the corresponding lot processing time, i.e.,

$$0 \leq T_{mr}^{N} \leq T_{mr} - 1, \quad \forall r, \forall m. \tag{7}$$

If the last time slot is not involved, every lot must be finished, therefore T_{mr}^{N} must be zero, i.e.,

$$\text{if } \delta_{mr}(K) = 0 \text{ then } T_{mr}^{N} = 0, \quad \forall r, \forall m. \tag{8}$$

The above constraint is logical, but it is easier to be linearized together with (7) as follows:

$$0 \leq T_{mr}^{N} \leq \delta_{mr}(K) \times (T_{mr} - 1), \quad \forall r, \forall m. \tag{9}$$

If $\delta_{mr}(K) = 0, 0 \leq T_{mr}^{N} \leq 0, T_{mr}^{N}$ must be zero; if $\delta_{mr}(K) = 1, 0 \leq T_{mr}^{N} \leq T_{mr} - 1$. Therefore, the set of linear constraints (9) satisfies both constraints (7) and (8) above.

C. Maximal Number of Lots Scheduled Constraints

If there is extra capacity beyond the total target, machines and reticles will be scheduled to process layers with high priorities because more reward is assigned as will be discussed in the objective function. However, this may lead to imbalance among layers of the same product because of the layer by layer process. To avoid this, the number of lots with layer l of product p to be processed should be under its upper bound, i.e.,

$$N_{pl} \leq T_{pl}^{UB}, \quad \forall p, \forall l. \tag{10}$$

In the above, T_{pl}^{UB} is the upper bound for layer l of product p, and this set of parameters are calculated offline based on heuristic rules (e.g., 1.2 times of target T_{pl}). The number of lots with layer l of product p to be processed within the day is denoted by N_{pl}, and this integer dependent variable can be derived from N_{mr} as follows:

$$N_{pl} = \sum_{m} \sum_{r \in S_{pl}^{R}} N_{mr}. \tag{11}$$

In the above, S_{pl}^{R} denotes the set of reticles that process layer l of product p.

D. Setups-Related Constraints

When one litho machine switches from processing one layer to another layer, the machine and a corresponding reticle need to be set up. Since one reticle can only process one particular product\layer, a layer process switch on the litho machine can be treated as a reticle switch. In addition, a setup time is generally much shorter than the corresponding lot processing time, the time of setup is ignored and the number of setups is considered. In this way, the number of resource setups can be modeled as the number of reticle changes. The key issue here is how to find the beginning and completion points of machine and reticle combinations. The two situations of one machine completes combining with one reticle and begins to combine with another reticle are shown in Fig. 2.

It can be seen that when the values of $\delta_{mr}(k)$ and $\delta_{mr}(k+1)$ switch from 1 to 0, machine m completes the combination with reticle r; when the values switch from 0 to 1, machine m begins the combination with reticle r. To get a linear formulation, a new set of binary decision variables $\{y_{mr}(k)\}$ is used as follows:

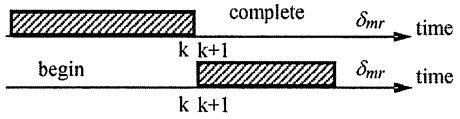


Fig. 2. Completion and beginning points of processing.

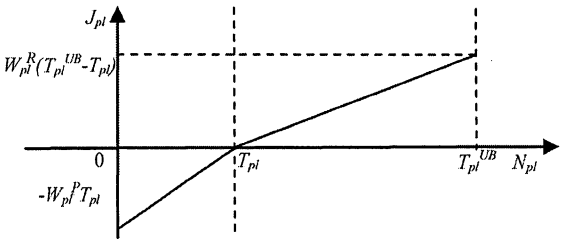


Fig. 3. The meeting target term of the objective function.

$$y_{mr}(k+1) \geq \delta_{mr}(k+1) - \delta_{mr}(k),$$
$$y_{mr}(k+1) \geq \delta_{mr}(k) - \delta_{mr}(k+1), \forall m, \forall r, 1 \leq k \leq K-1. \tag{12}$$

Since the purpose is to reduce the number of setups and $\{y_{mr}(k)\}$ is only shown in objective function, when $\delta_{mr}(k+1) - \delta_{mr}(k) = \pm 1$, $y_{mr}(k+1) = 1$; when $\delta_{mr}(k+1) - \delta_{mr}(k) = 0$, $y_{mr}(k+1) = 0$.

The beginning and completion points occur in pairs, the second half of (12) is therefore used for practical problems.

E. Objective Function

The objective function has four terms, to meet targets, balance future load, avoid simultaneous reticle expirations, and avoid excessive setups as presented next.

1) Meeting Targets: If machine assignments exceed the target, it is expressed as certain reward in the objective function; if not, it is expressed as corresponding penalty. Since different layers have different priorities, different weights are assigned. Let W_{pl}^{R} and W_{pl}^{P} denote the reward and penalty weights for layer l of product p. To check whether the assignments meet the daily target, a piecewise function is used, as shown in Fig. 3.

The upper bound and lower bound of N_{pl} is T_{pl}^{UB} and 0, and three break points are 0, T_{pl} and T_{pl}^{UB}. Based on the special ordered set techniques [8], the above formulation can be fully linearized.

2) Future Load Balancing: During lithography, a selected set of layers must be processed on the same machine for precision fabrication. For example, stacking layers A, B, and C (a stacking group) must be processed on the same machine, as shown in Fig. 4. If Machine 1 is assigned to process layer A within the day, those lots will come to Machine 1 for layers B and C in the future.

To avoid overload or starvation, the load on machines should be balanced. This is importance in view of the reentrant nature and the presence of stacking layers. Of particular interest is to balance the future stacking layer load for a specific day, e.g., the

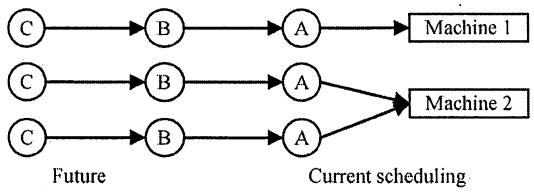


Fig. 4. Stacking layers.

day that the next layer will most likely to come back. Consider an example for a stacking group with layers A and B, and layer A is to be scheduled. The date that layer B will come back is probabilistic. This cycle time distribution can be obtained from historical date. Then, the expected load for the day with the highest probability that layer B will come back can be calculated based on the current and past assignments. One of the performance measures is the load difference between a machine load and the average load for layer B. The above will be made specific next.

Let WIP_{md}^{A-B} denote the number of lots whose layer A was processed d days ago on machine m, with the associated stacking layer B to be processed after a cycle time on the same machine. Let $P(CT_d^{A-B})$ denote the probability that the cycle time is d days from layers A to B. The total future load LT_m^{A-B} with layer B to be processed in the Tth day on machine m can be calculated as follows:

$$LT_m^{A-B} = N_m^A \times P\left(CT_T^{A-B}\right) + \sum_d WIP_{md}^{A-B} \times P\left(CT_{d+T}^{A-B}\right). \tag{13}$$

In the above, N_m^A denotes the number of lots whose layer A is scheduled to be processed on machine m within the day, T denotes the cycle time from layers A to B with the largest probability. Then, let LA^{A-B} denote the average load of machines for layer B. Load difference LD_m^{A-B} between total future load LT_m^{A-B} and the average LA^{A-B} on machine m can be described as follows:

$$LD_m^{A-B} = LT_m^{A-B} - LA^{A-B}. \tag{14}$$

In the above, LA^{A-B} is approximated based on WIP and the target of layer A, it therefore contains no decisions. Summation of the load difference is minimized as a part of the objective function.

3) Reticle Expiration: Reticles need to be recalibrated after processing a certain number of lots. Reticle remaining lifetime is used here to measure how many lots one reticle can process before next recalibration. Ideally, the expiration dates of reticles in the same group should be equally spaced with the same time interval to avoid simultaneous reticle expirations. The remaining lifetime difference between two reticles can be adjusted as the expected time interval selected based on heuristic rules within every reticle group through proper reticle assignments. Let R_r^0 denote the remaining lifetime of reticle r before scheduling, representing how many lots reticle r can process before next recalibration. Similarly, let R_r denote the remaining lifetime of reticle r after scheduling. Their relationship can be easily obtained from

$$R_r = R_r^0 - \sum_m N_{mr}. \tag{15}$$

If the remaining lifetime difference between two reticles is larger than the expected interval after scheduling, it is expressed as certain reward in the objective function; if not, it is expressed as corresponding penalty. This term is linearized similarly to the first term. Since only the difference of two reticles that have the closest remaining lifetimes is reasonable and useful, a sequence will be established for every reticle group based on their remaining lifetimes. Each reticle will be assigned a ranking number, the smaller the number, the longer the remaining lifetime. An important assumption here is that the rank does not change before and after scheduling.

4) Summary: The modeling of the number of resource setups has already been discussed in Section III-D, and the total number of setups is considered in the objective function.

In sum, the objective function with the above four terms to be minimized is described as follows:

$$\begin{aligned}
&\sum_p \sum_l \left(-W_{pl}^P \times \min(N_{pl} - T_{pl}, 0) - W_{pl}^R \right. \\
&\qquad \left. \times \max(N_{pl} - T_{pl}, 0)\right) \\
&\quad + W^L \times \sum_{g \in S^{SG}} \sum_{m \in S_g^{MS}} |LD_{gm}| \\
&\quad + \sum_p \sum_l \sum_{r_1 \in S_{pl}^R, r_2 \in S_{pl}^R, r_1 \neq r_2, r_2.No - r_1.No = 1} \\
&\quad \times \left(-W^{RP} \times \min\left(R_{r_1} - R_{r_2} - G_{pl}, 0\right) - W^{RR} \right. \\
&\qquad \left. \times \max\left(R_{r_1} - R_{r_2} - G_{pl}, 0\right)\right) \\
&\quad + W^R \times \sum_m \sum_r \sum_k y_{mr}(k).
\end{aligned} \tag{16}$$

In the second term, W^L denotes the weight for future stacking layer load balancing, S^{SG} denotes the set of stacking groups, and S_g^{MS} denotes the set of machines in stacking group g. In the third term, G_{pl} denote the expected expiration interval for reticles that process layer l of product p, and W^{RR} and W^{RP} denote the reward and penalty weights. In the last term, W^R denote the weight for avoiding excessive resource setups. The absolute values can be linearized similarly to (12). The above formulation is linear, and max and min are kept here for simplicity. The problem formulated is believed to be NP hard.

IV. Solution Methodology

The problem is solved by using the branch-and-cut method by exploiting problem linearity after simplification as presented in Section IV-A. Near-optimal solutions for practical problems, however, are still difficult to obtain efficiently as compared with the required time. The reason is that the convex hull of the problem is hard to delineate as explained with a small example in Section IV-B. To improve computation efficiency, a two-phase approach is therefore developed in Section IV-C. The convex hull of the first phase is analyzed with the same small example in Section IV-D.

A. Branch-and-Cut Method

The problem is solved by using the branch-and-cut method. Mixed-integer linear programming problems are usually difficult to solve because a set of decision variables are restricted to integer values. Branch-and-cut is powerful for certain classes of

mixed-integer linear optimization problems, and is easy to code by using commercial solvers. In the method, the integrality-relaxed problem is solved first by using a linear programming method. If all integer decision variables are integers, the solution is optimal to the original problem. If not, valid cuts that do not cut off any feasible integer solutions are added trying to obtain the convex hull (the smallest convex set that contains all feasible integer solutions in the Euclidean space). The idea is that once the convex hull is obtained, all integer decision variables of the linear programming solution are integers and optimal to the original problem. The process of obtaining the convex hull, however, is problem dependent, and can itself be NP hard. Low-efficient branching operations may then be needed on the variables whose values in the optimal relaxed solution violate their integrality requirements. The objective value of current optimal relaxed solution is a lower bound, and can be used to quantify the quality of a feasible solution. The optimization stops when CPU time reaches the preset stop time or the relative gap falls below the preset stop gap [11].

For the problem formulated above, although it is linear, the convex hull is still difficult to obtain. Processing time requirements with multiple decision variables $(\delta_{mr}(k), N_{mr}, T^N_{mr})$ might increase the difficulty of obtaining the convex hull because of complicating interactions among decisions. To overcome this, T^N_{mr} is removed, and two sets of decisions are left. Then, by relaxing the integrality requirements on N_{mr}, the processing time requirements (5) are modified as follows:

$$N_{mr} = \left(\sum_k \delta_{mr}(k) - T^P_{mr} \right) \div T_{mr}, \quad \forall r, \forall m. \tag{17}$$

In the above, N_{mr} is not a decision variable and may not be an integer. However, the integer part of N_{mr} still represents the number of lots scheduled on machine m and reticle r within the day, and the remaining fractional part can be used to derive T^N_{mr}. Optimality is therefore not affected. Based on the scheduling results, N_{mr} might need to be adjusted manually.

B. Convex Hull Analyses for the One-Phase Model

After the simplification above, near-optimal solutions for practical problems, however, are still difficult to obtain efficiently as will be shown in Section V. To overcome this difficulty, the convex hull is analyzed here. The problem now has three major sets of constraints: resource capacity, maximal number of lots scheduled and setup-related constraints since the processing time requirements were simplified as expressions (18). With $\delta_{mr}(k)$ represented by x_j $(1 \leq j \leq n = M \times R \times K)$, resource capacity constraints (1)–(4) can be expressed as $\sum_j a_j x_i \leq a_0$, where a_0 and a_j are positive integers and $x_j = 0$ or 1. Each of these constraints is a facet of the convex hull based on the proof in [5] and [6]. Through detailed analyses, it is discovered that the difficulty of obtaining the convex hull is caused by the setup-related constraints. To demonstrate this, convex hulls of the problems without and with these setup-related constraints are analyzed and compared through a simple example in this subsection.

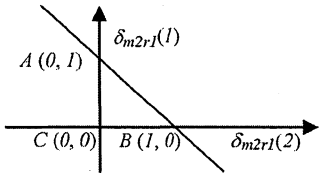


Fig. 5. The feasible region and convex hull of the problem without setup-related constraints.

Consider a simple example with two machines and two reticles. The first two sets of constraints mentioned above are considered in the first problem, and all of the three sets are considered in the second problem. Through the analyses and comparison of the two convex hulls, it is discovered that if the polyhedron formulated by all the constraints is simple and the convex hull can be easily obtained from the polyhedron, the problem can be efficiently solved; otherwise, low-efficient branching operations are needed. For visualization purpose, certain decision variables are fixed to present this intuitively.

In this example, two layers (l_1 and l_2) are to be scheduled on two machines (m_1 and m_2) with two reticles (r_1 for l_2 and r_2 for l_1). For simplicity, the total number of time slots is two and the processing time of both layers is one time slot. The targets are two lots for the first layer and one lot for the second layer.

In the first problem, the objective function is to meet targets. The problem is solved by using branch-and-cut. It is hard to visualize the convex hull and the optimal solutions because of the high dimensionality. For visualization purpose, $\delta_{m2r1}(1)$ and $\delta_{m2r1}(2)$ are selected to plot 2-D Fig. 5 with other decision variables fixed at their values in the optimal solution. An optimal solution is A (0, 1) (or B), which can be directly obtained from the convex hull ABC.

Since the future stacking layer load balancing and reticle expiration related constraints are only shown in the objective function, practical problems without setup-related constraints and term in the objective function can be efficiently solved.

In the second problem with setup-related constraints (12), the objective function is to meet targets and avoid excessive resource setups. The problem is also solved by using branch-and-cut, and $\delta_{m2r1}(k)$ and y_{m2r1} are selected to plot 3-D Fig. 6. After relaxing the integrality requirements, all decision variables can take any value within [0, 1], and the optimal relaxed solution is D (0.5, 0.5, 0). All constraints formulate this polyhedron ACBD, and the convex hull ABC cannot be obtained by adding cuts on the feasible region. This difficulty is caused by the interactions among decisions in the setup-related constraints. It can be seen that two of the three values in the optimal relaxed solution are non-integers. To get the optimal solution A (1, 0, 1) (or C), low-efficient branching operations need to be performed on the first two variables of the relaxed optimal solution.

Generally, for a problem with M machines, R reticles, and K time slots, the total number of setup-related constraints is $2MR(K-1)$. All constraints formulate a complicating polyhedron, and the convex hull is difficult to obtain. Because of these complicating setup-related constraints, near-optimal solutions cannot be efficiently obtained.

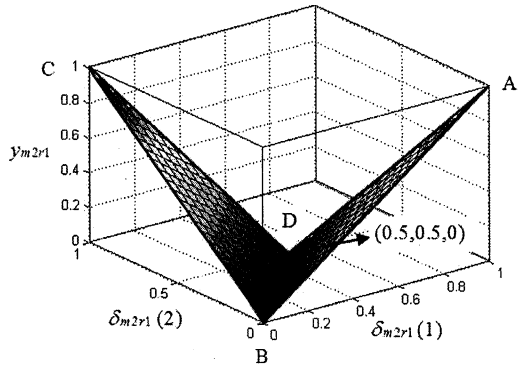


Fig. 6. The feasible region and convex hull of the problem with setup-related constraints.

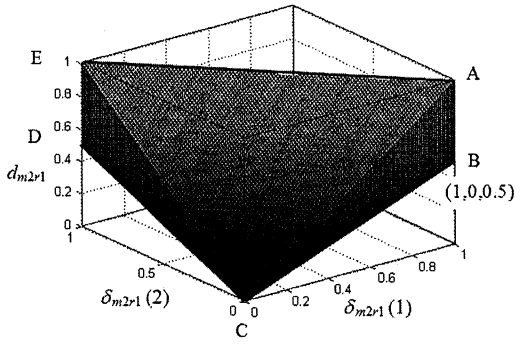


Fig. 7. The feasible region and convex hull for the first phase of the two-phase model.

C. A Two-Phase Approach

Based on the above analyses, a two-phase approach is developed to improve computation efficiency. In the first phase, a simplified problem without the complicating constraints is efficiently solved to establish ranges of decision variables. The problem with the full set of constraints is then solved in the second phase with a much reduced decision space.

In the first phase, the setup-related constraints are removed. After taking the setup-related term out of the objective function, many reticles might be assigned to a machine, leading to many excessive resource setups. To avoid this, the total number of reticles assigned is minimized as a part of the objective function. To check whether reticle r is assigned to machine m within the day, a new set of binary variables $\{d_{mr}\}$ is used

$$d_{mr} \leq N_{mr} \leq N_{mr}^{UB} \times d_{mr}, \quad \forall m, \forall r. \tag{18}$$

In the above equation, if $d_{mr} = 1$, reticle r is assigned to machine m; if $d_{mr} = 0$, otherwise. The last term in the objective function (16) needs to be revised correspondingly. The main decision variables are still $\delta_{mr}(k)$. Since there could be many excessive setups, $\delta_{mr}(k)$ cannot tell when the layers should be processed. Dependent variables N_{mr} are used as schedules instead.

The problem formulated above is solved by using branch-and-cut by exploiting linearity to reduce ranges of decision variables, and the solutions can be obtained fast as will be shown in Section V.

In the second phase, the ranges of decision variables are reduced by either fixing certain variables or by restricting the ranges of others based on first phase results. For example, assignments of machines and reticles to layers are fixed. In doing so, optimality might be affected as will be discussed in Section V. As another example, the range of the number of lots scheduled on a machine with a certain reticle is restricted. The magnitude of a range is selected based on testing results as a tradeoff between solution quality and computation efficiency. If this range is large enough, optimality will not be affected.

The objective function is the same as that of the one-phase approach. The decisions are $\delta_{mr}(k)$ (only for machines and reticles assigned in the first phase), which mean when the layer should be processed.

The problem formulated above is solved by using branch-and-cut. During branching operations, computation time is reduced in a major way since the ranges of decision variables are much reduced. The problem can therefore be solved faster than the one-phase approach as will be shown in Section V.

D. Convex Hull Analyses for the Two-Phase Model

To compare with the one-phase approach, the same simple example is analyzed. In the first phase, the problem is formulated with resource capacity (1) and (2), maximal number of lots scheduled (10) and (11), and reticle assigned detection constraints (18). The objective is to meet targets and reduce the number of reticles assigned. It is discovered that the number of non-integer values (=1) in the optimal relaxed solution is smaller than that (=2) of the one-phase approach, implying fewer branching operations and faster termination. In the second phase, the problem is formulated with resource capacity (1) and (2), setup-related (12), and variable range restriction constraints. The objective is to meet targets and avoid excessive resource setups. This problem can be efficiently solved because the decision space (=6) is smaller than that (=12) of the one-phase approach.

The problem is solved by using branch-and-cut. To compare with the one-phase approach, $\delta_{m2r1}(k)$ and d_{m2r1} are selected to plot 3-D Fig. 7. For the relaxed problem, all feasible solutions are in the polyhedron ABCDE, and an optimal relaxed solution is B (1, 0, 0.5) (or D). The convex hull ACE cannot be obtained by adding cuts on the feasible region, and an optimal solution is A (1, 0, 1) (or E). It can be seen only d_{m2r1} is a non-integer in the optimal relaxed solution and branching operation needs to be performed only on one variable.

In the second phase, based on the results from the first phase, the number of decisions is 6 ($\delta_{m2r1}(k)$, $\delta_{m1r2}(k)$, y_{m2r1} and y_{m1r2}) as compared with 12 in the one-phase model. The problem can be therefore solved faster than the one-phase approach.

For the problem with M machines, R reticles, and K time slots mentioned in Section IV-B, the number of reticle assigned related constraints (18) in the first phase is only MR as compared with $2MR(K-1)$ (12) in the one-phase model. The polyhedron formulated by constraints in the first phase is simpler than that of the one-phase model. The total number of branching operations needed is smaller, and the problem can be

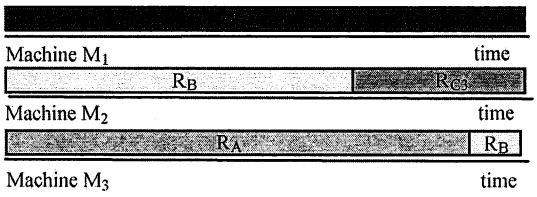


Fig. 8. Gantt chart of schedule results.

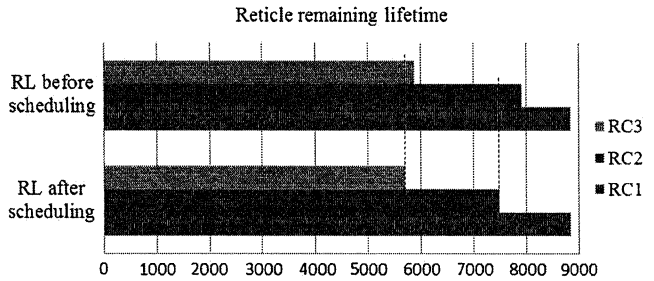


Fig. 9. Reticle remaining lifetime results.

efficiently solved to establish ranges of decision variables. In the second phase, the problem can be efficiently solved with a much reduced decision space. To quantify the solution quality, lower bounds should be obtained from the one-phase approach since the objective values of optimal relaxed solutions of the two-phase approach might not be lower bounds to the original problem. For this simple example, the final gap turns out to be zero, implying that the optimal solution has been obtained.

V. Numerical Results

The methods presented above have been implemented by using the optimization package IBM ILOG CPLEX Optimization Studio V 12.2. Testing has been performed on a PC with 1.60 GHz Intel (R) i7 CPU and 4 G RAM, and three examples are presented. The first small example is to demonstrate schedules of small problems obtained by using the one-phase approach can be duplicated and obtained by hands. The second medium-sized example is to compare the statistics of one-phase and two-phase approaches. The third practical example is to compare solution quality and computation efficiency of these two approaches, demonstrating that the two-phase approach can generate near-optimal schedules within much reduced computation time.

Example 1: Testing of the One-Phase Approach With a Small Problem: This small example is to demonstrate schedules of small problems obtained by using the one-phase approach can be duplicated and obtained by hands. In this example, three layers of one product are to be scheduled on three machines with five reticles in 102 time slots. The information is as follows.

- Layers: L_A and L_B (stacking group), and L_C.
- Machines: M_1, M_2 and M_3 with total future stacking layer load of 6983, 4490, and 2044, respectively.
- Reticles: R_A, R_B, R_{C1}, R_{C2}, and R_{C3} with remaining lifetime of 89572, 88308, 8442, 79059, and 58732, respectively.
- Target: 600 for L_A, 650 for L_B and 600 for L_C.

The problem is solved in 3 seconds with a relative gap of 5% by using the one-phase approach, and the Gantt chart is shown in Fig. 8.

The schedule meets all targets of three layers without excessive resource setups. It can be seen that layer L_A is all scheduled on Machine M_3 since it has lowest total load before scheduling, and it also shows the stacking layer load will be balanced in the next few days. In addition, the remaining lifetime difference between R_{C1} and R_{C2} becomes larger, as shown in Fig. 9, which avoids simultaneous expirations of the three reticles that process layer L_C. The above schedule can be duplicated and obtained by hand, and also shows that our method can satisfy the objective of this litho machine scheduling problem.

TABLE I
Testing Results Of Example 2

	One-phase model	The first phase of two-phase model	The second phase of two-phase model
CPU time	123.03 s	10.02 s	4.77 s
Relative gap	4.32%	0.45%	0.47%
Number of var	19,951	10,051	2,031
Number of cons	22,904	3,104	2,352
Objective value	837	852	
Final relative gap	4.32%	6.19%	
Node processed	16	31	10
Root relaxation time	3.12 s	0.14 s	0.11 s
Number of cuts	1	241	368
Time on cuts	29.5 s (24%)	4 s (40%)	0.3 s (6%)
Time on branching	87.4 s (71%)	4.5 s (45%)	2.6 s (55%)

Example 2: Testing of the One-Phase and Two-Phase Approaches With a Medium-Sized Problem: This medium-sized example is to compare the statistics of one-phase and two-phase approaches. In this example, four layers of one product are to be scheduled on five machines with ten reticles in 200 time slots. For simplicity, the objective function is to meet targets and avoid excessive resource setups in the one-phase model and in the second phase of the two-phase model; to meet targets and reduce the number of reticles assigned in the first phase of the two-phase model.

The testing results are shown in Table I.

From the results, it can be seen that the number of variables in the first phase of the two-phase model is around 44% of that in the one-phase model, and the number of constraints is about 14%. In the second phase, the numbers of variables and constraints are about 10% of those in the one-phase model. In addition, the time on branching is 4.5 s and 2.6 s in the first and second phases of the two-phase model, as compared with 87.4 s in the one-phase model. The total CPU time on preprocessing, cutting and branching required by the two-phase approach is 15 s as compared with 123 s required by the one-phase approach. Although the final gap is about 2% higher than that of the one-phase approach, the computation efficiency is much improved. Both of the scheduling results meet the targets, and there are five and six setups in the one-phase and two-phase results, respectively.

Example 3: Testing of the One-Phase and Two-Phase Approaches With a Practical Problem: This practical problem is

TABLE II
TESTING RESULTS OF EXAMPLE 3

	One-phase model	The first phase of two-phase model	The second phase of two-phase model
Stop time	300 s	120 s	240 s
Stop gap	5 %	0.1%	0.5%
CPU time	279 s	25 s	2 s
Relative gap	4.3%	0.1%	0.1%
Objective value	86362.5	87281.2	
Final relative gap	4.3%	5.4%	

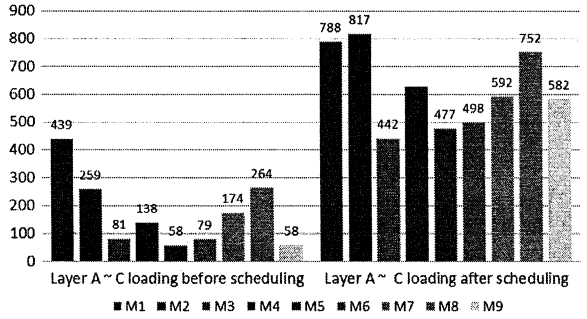


Fig. 10. Future stacking layer load results.

to compare solution quality and computation efficiency of one-phase and two-phase approaches. In this example, seven layers of one product are to be scheduled on 11 machines with 71 reticles in 411 time slots (one day). The objective function of the second phase here is to increase the number of lots processed, finish targets as soon as possible, and reduce the number of reticles assigned, and reduce the number of resource setups. For simplicity of implementation, future stacking layer load balancing and reticle expiration terms are removed as they have already been mostly satisfied in the first phase. Because of this, the final cost is obtained by plugging the solution into the original objective function (16).

The testing results are shown in Table II.

This practical problem is solved in 5 minutes with a gap of 4.3% by using the one-phase approach. This computation time is still long as compared to the required time, 2 minutes. By using the two-phase approach, the same problem is solved within 30 s. Although the final gap is 5.4% and 1.1% higher than that of the one-phase approach, it is still acceptable. The schedule obtained from the two-phase approach meet all targets without excessive resource setups except for one layer, because there are no available reticles for this layer. In terms of future stacking layer load balancing, the total load among machines from stacking layers A–C before and after scheduling is compared in Fig. 10. It can be seen that the future load is nearly balanced through scheduling. In addition, the remaining lifetime difference between two reticles that have the closest remaining lifetimes in the same group is moving toward the expected interval after scheduling, which avoids simultaneous reticle expirations.

VI. CONCLUSION

In this paper, a novel mathematical formulation for litho machine scheduling over a day with resource setups, reticle expirations and future stacking layer load balancing is established. The problem is solved by using branch-and-cut by exploiting problem linearity. To improve computation efficiency, a two-phase approach is developed. In the first phase, a simplified problem with certain complicating constraints dropped is efficiently solved to establish ranges of decision variables. The problem with the full set of constraints is then solved in the second phase with a much reduced decision space. Numerical testing shows that the two-phase approach generates near-optimal schedules within reasonable amounts of computation time.

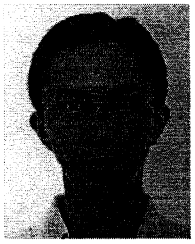
With minor changes in the formulation, our method is also used for real-time rescheduling every 10 minutes. Furthermore, this two-phase approach is generic, and will have major implications on other semiconductor scheduling problems and beyond.

REFERENCES

[1] E. Akcali, K. Nemoto, and R. Uzsoy, "Cycle-Time improvements for photolithography process in semiconductor manufacturing," *IEEE Trans. Semiconductor Manuf.*, vol. 14, no. 1, pp. 48–56, 2001.
[2] E. Akcali and R. Uzsoy, "A sequential solution methodology for capacity allocation and lot scheduling problems for photolithography," in *Proc. 26th IEEE/CPMT Int., Electron. Manuf. Technol. Symp.*, Santa Clara, CA, USA, 2000, pp. 374–381.
[3] E. Akcali, A. Üngör, and R. Uzsoy, "Short-term capacity allocation problem with tool and setup constraints," *Naval Res. Logistics*, vol. 52, no. 8, pp. 754–764, 2005.
[4] A. Arisha and P. Young, "Intelligent simulation-based lot scheduling of photolithography toolsets in a wafer fabrication facility," in *Proc. 36th Conf. Winter Simulation*, 2004, pp. 1935–1942.
[5] E. Balas, "Facets of the knapsack polytope," *Math. Program.*, vol. 8, no. 1, pp. 146–164, 1975.
[6] E. Balas and E. Zemel, "Facets of the knapsack polytope from minimal covers," *SIAM J. Appl. Math.*, vol. 34, no. 1, pp. 119–148, 1978.
[7] F. Barahona, S. Bermon, O. Günlük, and S. Hood, "Robust capacity planning in semiconductor manufacturing," *Naval Res. Logistics*, vol. 52, no. 5, pp. 459–468, 2005.
[8] E. M. L. Beale and J. J. H. Forrest, "Global optimization using special ordered sets," *Math. Program.*, vol. 10, no. 1, pp. 52–69, 1976.
[9] S. C. Chang and D. Y. Liao, "Scheduling flexible flow shops with no setup effects," *IEEE Trans. Robot. Autom.*, vol. 10, no. 2, pp. 112–122, Apr. 1994.
[10] J. Chung and J. Jang, "A WIP balancing procedure for throughput maximization in semiconductor fabrication," *IEEE Trans. Semiconductor Manuf.*, vol. 22, no. 3, pp. 381–390, Aug. 2009.
[11] *"IBM, ILOG CPLEX 12.2 User's Manual,"* 2010.
[12] Y. H. Lee, J. Park, and S. Kim, "Experimental study on input and bottleneck scheduling for a semiconductor fabrication line," *IIE Trans.*, vol. 34, no. 2, pp. 179–190, 2002.
[13] D. Y. Liao, S. C. Chang, K. W. Pei, and C. M. Chang, "Daily scheduling for R&D semiconductor fabrication," *IEEE Trans. Semiconductor Manuf.*, vol. 9, no. 4, pp. 550–561, 1996.
[14] L. Mönch, M. Prause, and V. Schmalfuss, "Simulation-based solution of load-balancing problems in the photolithography area of a semiconductor wafer fabrication facility," in *Proc. 33rd Conf. Winter Simulation*, 2001, pp. 1170–1177.
[15] S. C. Sarin, L. Wang, and M. Cheng, "A single-machine, single-wafer-processing, multiple-lots-per-carrier scheduling problem to minimize the sum of lot completion times," *Comput. Oper. Res.*, vol. 39, no. 7, pp. 1411–1418, 2012.
[16] A. M. D. Shr, A. Liu, and P. P. Chen, "Load balancing among photolithography machines in the semiconductor manufacturing system," *J. Inform. Sci. Eng.*, vol. 24, no. 2, pp. 379–391, 2008.
[17] L. B. Toktay and R. Uzsoy, "A capacity allocation problem with integer side constraints," *Eur. J. Oper. Res.*, vol. 109, no. 1, pp. 170–182, 1998.
[18] F. D. Vargas-Villamil and D. E. Rivera, "Multilayer optimization and scheduling using model predictive control: Application to reentrant semiconductor manufacturing lines," *Comput. Chem. Eng.*, vol. 24, no. 8, pp. 2009–2021, 2000.
[19] J. Wang and P. B. Luh, "Scheduling of a machining center," *Math. Comput. Modeling on Recent Adv. Discrete Event Syst.*, vol. 23, no. 11-12, pp. 203–214, 1996.
[20] C. Yugma, R. Riffart, S. Dauzere-Peres, P. Vialletelle, and F. Buttin, "A dispatcher simulator for a photolithography workshop," in *Proc. IEEE/SEMI Adv. Semiconductor Manuf. Confe.*, Stresa, Italy, 2007, pp. 100–104.

Bing Yan (S'11) received the B.S. degree from Renmin University of China, Beijing, China, in 2010. She is currently working towards the Ph.D. degree at Department of Electrical and Computer Engineering, University of Connecticut, Storrs, CT, USA.

Her research interests include scheduling of manufacturing systems, optimized energy management of smart buildings and eco-communities.



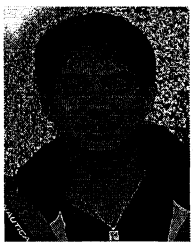
Hsin Yuan Chen received the B.S. degree from National Tsing Hua University, Hsinchu, Taiwan, in 2002 and the M.S. degree from the University of Florida, Gainesville, FL, USA, in 2009.

He is working as the Engineer at Inotera Memories Inc. His interest includes optimization in scheduling and linear and integer programming.

Peter B. Luh (S'77–M'80–SM'91–F'95) received the B.S. degree from National Taiwan University, Taipei, the M.S. degree from the Massachusetts Institute of Technology (MIT), Cambridge, MA, USA, and the Ph.D. degree from Harvard University, Cambridge, MA, USA.

He is the SNET Professor of Communications and Information Technologies at the University of Connecticut, and a member of the Chair Professors Group at CFINS in Tsinghua University, Beijing, China. He

Prof. Luh was the founding Editor-in-Chief of the IEEE TRANSACTIONS ON AUTOMATION SCIENCE AND ENGINEERING (T-ASE) and a former Editor-in-Chief of the IEEE TRANSACTIONS ON ROBOTICS AND AUTOMATION. His interests include intelligent manufacturing systems; smart, green and safe buildings; and smart power systems.



Simon Wang (S'11) received the B.S. degree in industry engineering from Chung Yuan Christian University, Chung Li, Taiwan, in 1999 and the M.S. degree in industry engineering and management from National Yunlin University of Science and Technology, Yunlin, Taiwan.

He is currently working at the Kaizen and Lean Manufacturing Department, Inotera Memories Inc. His research interests include scheduling of manufacturing systems, WIP flow strategy and real time dispatching logistic.

Joey Chang, photograph and biography not available at the time of publication.