This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

Solving an Extended Double Row Layout Problem Using Multiobjective Tabu Search and Linear Programming

Xingquan Zuo, *Member, IEEE*, Chase C. Murray, and Alice E. Smith, *Senior Member, IEEE*

***Abstract*—Facility layout problems have drawn much attention over the years, as evidenced by many different versions and formulations in the manufacturing context. This paper is motivated by semiconductor manufacturing, where the floor space is highly expensive (such as in a cleanroom environment) but there is also considerable material handling amongst machines. This is an integrated optimization task that considers both material movement and manufacturing area. Specifically, a new approach combining multiobjective tabu search with linear programming is proposed for an extended double row layout problem, in which the objective is to determine exact locations of machines in both rows to minimize material handling cost and layout area where material flows are asymmetric. First, a formulation of this layout problem is established. Second, an optimization framework is proposed that utilizes multiobjective tabu search and linear programming to determine a set of non-dominated solutions, which includes both sequences and positions of machines. This framework is applied to various manufacturing situations, and compared with an exact approach and a popular multiobjective genetic algorithm optimization algorithm. Experimental results show that the proposed approach is able to obtain sets of Pareto solutions that are far better than those obtained by the alternative approaches.**

***Note to Practitioners*—Manufacturing facility layouts are typically designed with the singular focus of promoting minimum-distance material flows, as material handling costs increase with travel distance. However, the square footage required by a layout also contributes to manufacturing costs, as is especially true in cleanroom environments typically found in semiconductor manufacturing. This paper proposes a methodology for determining a set of Pareto optimal solutions, highlighting the tradeoff between minimizing material flow cost versus facility size. We demonstrate this approach on a commonly used layout configuration, where two rows of machines are separated by an aisle. Despite the straightforward problem description, determining optimal solutions to this problem is challenging. Our approach is shown to work efficiently on problems of realistic size and provide implementable machine layouts.**

***Index Terms*—Facility layout problems, linear programming, multiobjective optimization, tabu search (TS).**

Manuscript received November 14, 2013; accepted January 31, 2014. This paper was recommended for publication by Associate Editor H. Wang and Editor M. C. Zhou upon evaluation of the reviewers' comments. This paper was supported in part by the National Natural Science Foundation of China under Grant 61374204.

X. Zuo is with the Computer School, Beijing University of Posts and Telecommunications, Beijing 100876, China (e-mail: zuoxq@bupt.edu.cn; zuoxq@ieee.org).

C. C. Murray and A. E. Smith are with the Department of Industrial and Systems Engineering, Auburn University, Auburn, AL 36849 USA (e-mail: cmurray@auburn.edu; smithae@auburn.edu).

Digital Object Identifier 10.1109/TASE.2014.2304471

I. Introduction

THE FACILITY layout problem is well known and has been studied for decades (c.f., [1] and [2]). Traditionally, this problem has focused solely on minimizing the cost of material flow as a surrogate for minimizing material handling costs. However, for some industries and situations, the area is also an important factor to be considered. Such a situation has widespread application in high value manufacturing environments (e.g., cleanrooms), where material flow can be complex and construction costs are approximately \$3,500 per square foot or higher [3]. Unfortunately, minimizing material handling costs and minimizing layout area are often competing objectives. Furthermore, due to the vastly differing scales of these objectives, it may be difficult to combine them into a meaningful single objective. Instead, a multiobjective approach is proposed to determine the Pareto front of optimal solutions. The benefit of such an approach is to provide decision makers with the ability to choose the most appropriate layout that balances the tradeoffs between space consumption and material handling efficiency.

We propose a multiobjective approach to determine Pareto optimal solutions in the context of a double row layout problem (DRLP). The DRLP, first proposed by [4], seeks to determine the exact location for each machine in one of two rows (upper or lower) to minimize the total material handling cost. This problem requires the determination of not only the sequence of machines in each row (relative placement), but also of the actual location of each machine (absolute placement). The complexity of this problem is due to the incorporation of both combinatorial and continuous aspects. The DRLP has practical application in many industries because determining an exact location for each machine in both rows can result in a layout with less material handling cost than only considering sequence. Such a layout configuration is common in the bay/chase layouts often employed in semiconductor manufacturing.

Unfortunately, the standard DRLP focuses only on minimizing the cost of material flow, where the area consumed is ignored. Therefore, we define the extended DRLP (EDRLP) that addresses the impact of area consumption, an important factor for many industries. A new approach combining multiobjective tabu search (MTS) with linear programming (LP) is proposed for obtaining a set of Pareto solutions for the EDRLP. MTS is used to handle the combinatorial part of the EDRLP—the machine sequence—and linear programming is used to optimize the exact location of each machine on its assigned row. To improve the computational efficiency of MTS, a neighborhood reducing strategy is used to exclude those neighbors that cannot improve the quality of the current solution from the neighborhood set.

The remainder of this paper is organized as follows. A review of related facility layout problems is provided in Section II. The EDRLP and its formulation are presented in Section III. A characterization of EDRLP solutions is provided in Section IV, followed by details of the proposed MTS-LP in Section V. The effectiveness of MTS-LP is computationally verified through various problem instances with different sizes in Section VI. We conclude this paper with summary remarks in Section VII. An illustrative case, providing an annotated discussion for various layouts that may be produced by the MTS-LP approach, is provided as an online electronic supplement to this paper.

II. Related Literature

Research on facility layout problems is vast and spans several decades. Recent reviews of such problems may be found in [2], [5], and [6]. Among these problems, the single-, multiple-, and double row layout problems (SRLP, MRLP, and DRLP, respectively) and the corridor allocation problem (CAP) are most related to our work. Although there has been a fair number of papers on the SRLP (c.f., [7]–[11]), the MRLP (c.f., [12]–[14]), and the CAP (c.f., [15], [16]), to the best of our knowledge there exist only four published works devoted to the DRLP. The first mixed integer programming formulation appears in [4], along with five heuristics for the single objective of minimizing material flow cost. These heuristics were reported to produce solutions with large optimality gaps. These large gaps may be related to the discovery in [17] that the model in [4] allows layouts that ignore the minimum clearance requirements between some adjacent machines. A corrected formulation for the DRLP was provided by [17], although no solution approach was proposed. [18] proposed an optimal solution approach to a variant of the DRLP in which there are no minimum clearances between adjacent tools. This approach was tested on problems with up to 12 machines. In our previous work [19], an extended double row layout problem (EDRLP) was proposed in which nonzero aisle widths are allowed and the optimization objectives of both cost and layout area are considered. The two objectives were linearly combined to form a single objective, and CPLEX was used to solve this EDRLP. Unfortunately, the linear combination of the material handling and space consumption objectives creates challenges when these terms are of vastly different orders. The present work seeks to alleviate this issue by constructing a Pareto front of solutions related to both objectives.

III. The Extended Double Row Layout Problem (EDRLP)

As in the original DRLP formulation of [4], the EDRLP considers m machines to be allocated to two rows, where I represents the set of machines ($I = \{1, \ldots, m\}$) and R represents the set of rows ($R = \{1(\text{upper}), 2(\text{lower})\}$). Each pair of machines $i \in I^1$ and $j \in I^2$ must be separated by a minimum clearance, denoted as a_{ij}, where $I^1 = \{1, \ldots, m-1\}$ and $I^2 = \{i+1, \ldots, m\}$ for all $i \in I^1$. The material handling cost between pairs of machines is given by f_{ij}, which incorporates the flow frequency times the unit handling cost. Each machine $i \in I$ has a width w_i, and it is assumed that the load/unload port of each tool is located at the midpoint of the machine's width.

TABLE I
Decision Variables

x_{ir}	Continuous decision variable representing the location of machine $i \in I$ in row $r \in R$, such that $x_{ir} = 0$ if i is not placed in row r.
y_{ir}	Binary decision variable, such that $y_{ir} = 1$ if machine $i \in I$ is placed in row $r \in R$.
z_{rij}	Binary decision variable, such that $z_{rij} = 1$ if machine $i \in I$ is placed to the left of machine $j \in \{I \setminus i\}$ in row $r \in R$.
W	Width of the resulting layout. This is the maximum distance between the left side of the first machine in either row and the right side of the last machine in either row.
s_r	Area consumed by the machines in row $r \in R$.
A	Total area consumed by the resulting layout, as determined by the area of the smallest rectangle enclosing all machines (and the aisle between the rows). Clearance space for end machines of rows is not considered.
q_{ij}	Binary decision variable, such that $q_{ij} = 1$ if machines $i \in I^1$ and $j \in I^2$ are placed in the same row.
v_{ij}^+, v_{ij}^-	Auxiliary continuous decision variables employed to determine the absolute value of the distance between machines i and j.

The EDRLP extends the DRLP in two aspects. First, a nonzero width aisle between two rows is considered, denoted by c. Second, the objective of minimizing total layout area is considered along with the objective of minimizing material handling cost. The determination of total layout area requires knowledge of each machine's depth, d_i. A mixed integer programming formulation is established for the EDRLP as per [19]. In this formulation, we make the assumption that rectilinear travel for material flow goes across and along the aisle, however, material flow within the same row does not travel into the aisle. Decision variables for the EDRLP are defined in Table I.

The EDRLP has two objectives, namely the total cost of material flow and the layout area. The formulation below serves to minimize those two objectives simultaneously

$$\text{Minimize} \quad \{Obj_1, Obj_2\} \tag{1}$$

$$Obj_1 = \sum_{i \in I^1} \sum_{j \in I^2} (f_{ij} + f_{ji}) \left(v_{ij}^+ + v_{ij}^- + c(1 - q_{ij})\right) \tag{2}$$

$$Obj_2 = A \tag{3}$$

$$\text{subject to:} \quad x_{ir} \leq M y_{ir} \quad \forall i \in I, r \in R \tag{4}$$

$$\sum_{r \in R} y_{ir} = 1 \quad \forall i \in I \tag{5}$$

$$\frac{w_i y_{ir} + w_j y_{jr}}{2} + a_{ij} z_{rji} \leq x_{ir} - x_{jr} + M(1 - z_{rji}) \quad \forall i \in I^1, j \in I^2, r \in R \tag{6}$$

$$\frac{w_i y_{ir} + w_j y_{jr}}{2} + a_{ij} z_{rij} \leq -x_{ir} + x_{jr} + M(1 - z_{rij}) \quad \forall i \in I^1, j \in I^2, r \in R \tag{7}$$

$$\sum_{r \in R} x_{ir} - \sum_{r \in R} x_{jr} = v_{ij}^- - v_{ij}^+ \quad \forall i \in I^1, j \in I^2 \tag{8}$$

$$z_{rij} + z_{rji} \leq y_{ir} \quad \forall i \in I^1, j \in I^2, r \in R \tag{9}$$

$$z_{rij} + z_{rji} \leq y_{jr} \quad \forall i \in I^1, j \in I^2, r \in R \tag{10}$$

$$z_{rij} + z_{rji} + 1 \geq y_{ir} + y_{jr} \quad \forall i \in I^1, j \in I^2, r \in R \tag{11}$$

$$W \geq x_{ir} + \left(\frac{1}{2}\right) w_i y_{ir} \quad \forall i \in I, r \in R \tag{12}$$

$$x_{ir} - \left(\frac{1}{2}\right) w_i y_{ir} \geq 0 \quad \forall i \in I, r \in R \tag{13}$$

$$s_r \geq d_i W - d_i M(1 - y_{ir}) \quad \forall r \in R, i \in I \tag{14}$$
$$A = s_1 + s_2 + cW \tag{15}$$
$$q_{ij} = \sum_{r \in R} (z_{rij} + z_{rji}) \quad \forall i \in I^1, j \in I^2 \tag{16}$$
$$x_{ir} \geq 0 \quad \forall i \in I, r \in R \tag{17}$$
$$v_{ij}^+, v_{ij}^- \geq 0 \quad \forall i \in I^1, j \in I^2 \tag{18}$$
$$y_{ir} \in \{0, 1\} \quad \forall i \in I, r \in R \tag{19}$$
$$z_{rij} \in \{0, 1\} \quad \forall i \in I, j \in \{I \setminus i\}, r \in R \tag{20}$$
$$q_{ij} \in \{0, 1\} \quad \forall i \in I^1, j \in I^2 \tag{21}$$
$$s_r \geq 0 \quad \forall r \in R \tag{22}$$
$$A, W \geq 0. \tag{23}$$

The objective function (1) consists of two sub-objectives that are to be minimized simultaneously. The sub-objective in (2) describes the total weighted asymmetric material handling cost. This is the total unit cost of flow between machines i and j multiplied by the rectilinear distance between these machines (including the absolute value of the horizontal distance between machine pairs and the width of the aisle, c, in the event that the machines are in opposite rows). The sub-objective in (3) describes the overall area of the layout.

Constraints (4) and (5) ensure that each machine is placed in exactly one row. Constraints (6) and (7) guarantee that the minimum clearance between adjacent machines is satisfied. The absolute value of the horizontal distance between machines is determined by Constraint (8). Constraints (9)–(11) relate binary decision variables z_{rij} and y_{ir}, such that when machines i and j are both assigned to row r (i.e., $y_{ir} = y_{jr} = 1$), either z_{rij} or z_{rji} should be equal to 1; otherwise, $z_{rij} = z_{rji} = 0$. Constraints (12)–(14) are employed to determine lower bounds on the width (horizontal dimension) and area of a layout. The objective function term that seeks to minimize the total layout area, A, serves to make constraints (12) and (14) binding. Constraint (15) determines the total area of the resulting layout as the sum of the areas of row 1, row 2, and the corridor (aisle) separating the rows. Constraint (16) determines whether machines i and j are in the same row. Finally, (17)–(23) describe the decision variable definitions. The constant M is a sufficiently large number, and is given by

$$M = \sum_{i \in I} \left\{ w_i + \max_{\substack{j \in I \\ j \neq i}} (a_{ij}) \right\}.$$

As an extension of the DRLP, the EDRLP is NP-hard. For such problems, the computational time of any exact approach will increase exponentially with the number of machines, such that it is impossible to obtain an optimal solution within an acceptable computational time for a large size problem. In the next two sections, we describe a new solution approach that works well for problems of realistic size.

IV. Characterizing EDRLP Solutions

Solving the EDRLP requires not only determining the sequences of machines in both rows (binary decision variables), but also the exact locations of machines (continuous decision variables). Thus, if two layouts have the same sequences of machines but different machine locations, they are two different

Row 1	8	7	2	3	1
Row 2	9	6	4	5	10

Fig. 1. Representation of a discrete solution S.

solutions. For this reason, we draw the distinction between *discrete solutions* (defined by the sequence of machines in both rows) and *continuous solutions* (including the absolute location of each machine for a given sequence). Let S be defined to be a discrete solution, an example of which is shown in Fig. 1.

The distinction between discrete and continuous solutions is important in the proposed two-part approach for efficiently obtaining the Pareto front. Specifically, our approach employs multiobjective tabu search (MTS) to find the set of nondominated discrete solutions. For a particular discrete solution—whereby binary variables q_{ij}, y_{ir}, and z_{rij} in the formulation are fixed—finding optimal locations of machines is a pure linear programming (LP) problem, which is readily solved by LP software (e.g., CPLEX). The final Pareto solutions of EDRLP are generated from this set of non-dominated continuous solutions. The key issues of MTS-LP consist of: 1) how to determine non-dominated continuous solutions for a discrete solution; 2) how to produce the set of Pareto solutions from the set of non-dominated continuous solutions; and 3) how to find the set of non-dominated discrete solutions by the MTS. The first and second questions will be answered in this section and the third one in Section V.

A. Determining Nondominated Continuous Solutions Using Linear Programming

There are two special continuous solutions that may be constructed for a particular discrete solution S—one that minimizes the *area* and one that minimizes the *cost*. Recall that the area of a layout is defined to be the area of the smallest rectangle enclosing all machines. Given a sequence of machines, the depth dimension is determined by the deepest machine in each row. The minimum feasible width of a layout for a fixed sequence is found by separating adjacent machines by their minimum allowable clearances. We denote the minimal-area layout by continuous solution s_a, as shown in Fig. 2(a). The width and depth of s_a are expressed by W_a and D_a, respectively, and its area is $W_a \times D_a$.

While s_a is a continuous solution with minimal area, it may not have minimal material handling cost. The following method, termed *linear programming method 1* (LPM1), is designed to find such a solution quickly.

Step 1) The discrete solution S is used to fix all binary decision variables y_{ir}, z_{rij}, and q_{ij} in the formulation from Section III, such that the formulation becomes a pure linear program. We denote this LP as formulation P.

Step 2) Considering Obj_1 (cost) as the only objective function, CPLEX is used to solve P, obtaining values of all continuous decision variables (e.g., x_{ir}). This indicates the exact locations of machines in continuous space. A continuous solution s_b with minimal cost is shown in Fig. 2(b).

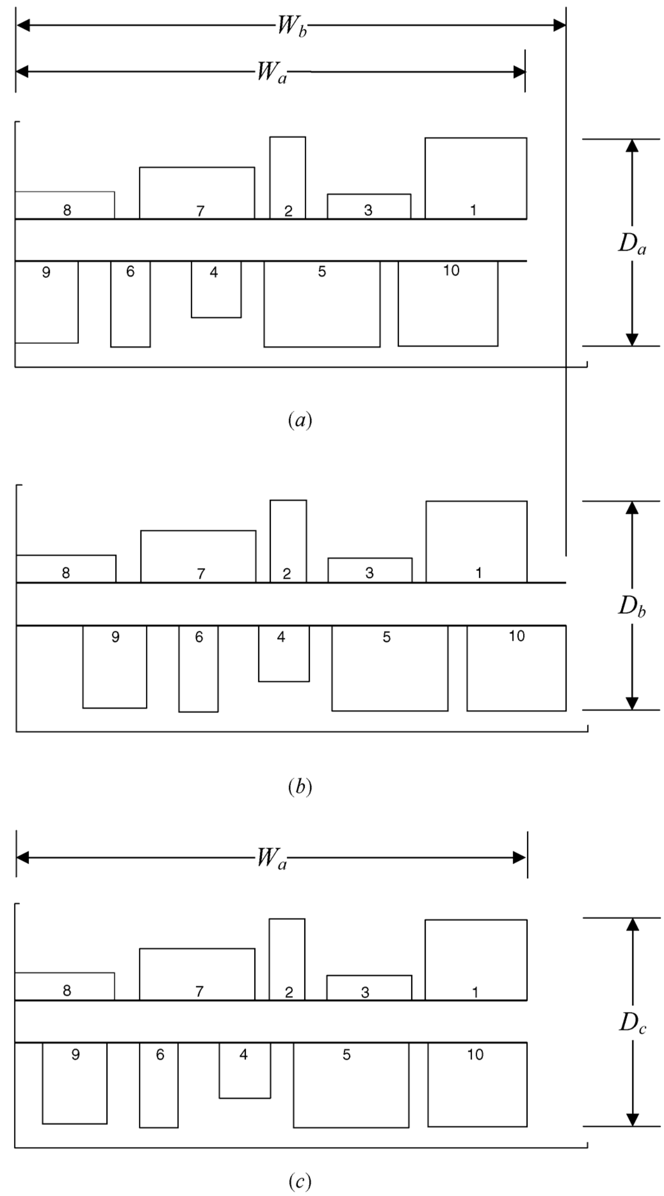

Fig. 2. A comparison of a minimal-area layout (continuous solution s_a), a minimal-cost layout (continuous solution s_b), and a minimal-cost layout subject to an area constraint (continuous solution s_c) for a particular discrete solution S.

It can be observed that s_b has a larger width (and, therefore, a larger area) than s_a, which means that reducing the material handling cost in this case requires additional clearances between some machines. At this point, we have identified two extreme continuous solutions, each of which minimizes exactly one of the two objectives for a given sequence of machines. Thus, we know that there is no continuous solution for S with a width less than W_a. Furthermore, we know that any continuous solution for S with a width greater than W_b must be dominated by s_b. Therefore, the width of non-dominated continuous solutions for a particular discrete solution S must be in the range $[W_a, W_b]$. Next, we describe a method for finding the set of all nondominated continuous solutions (denoted as $C(S)$) from the set of all continuous solutions ($AD(S)$) for a given discrete solution (S).

1) Constructing Pareto Continuous Solutions: While s_a is a solution with minimal width, another continuous solution of width W_a may also exist with a lower cost. To find such a solution, a constraint $A \leq D \times W_a$ is added to the formulation (where D is the depth of each solution in $AD(S)$). All binary decision variables are fixed according to the discrete solution S, and then linear programming is used to obtain a continuous solution s_c, as shown in Fig. 2(c). s_c is the one whose cost is minimal among the solutions with width W_a. From Fig. 2, it can be seen that s_a and s_c have the same width (W_a) but their absolute machine locations are different. Note that s_c dominates s_a (abbreviated as $s_c \succ s_a$), since they have the same area but s_c cannot have a greater cost.

As mentioned above, the width of a non-dominated continuous solution in $AD(S)$ must be in the range $[W_c, W_b]$ (or, equivalently, $[W_a, W_b]$). We have seen how to generate two of these continuous solutions—s_b (minimum cost) and s_c (minimum cost for the minimum area). Between them lie an infinite number of solutions comprising the set of non-dominated continuous solutions associated with a particular discrete solution S, denoted by $C(S)$.

Although it is impossible to identify every continuous solution in $C(S)$, we can modify the procedure used to generate s_c to find an approximation of $C(S)$. This procedure, termed *linear programming method 2* (LPM2), will determine the minimum-cost continuous solution for any given width $W \in [W_a, W_b]$ by relaxing the area constraint to reduce cost.

Step 1) The discrete solution S is used to fix all binary decision variables y_{ir}, z_{rij}, and q_{ij} from the formulation in Section III. The resulting LP is denoted as formulation P.

Step 2) Add the following area constraint to P and consider Obj_1 (cost) as the only objective function, obtaining the formulation P_W:

$$A \leq D \times W$$

where A is the decision variable for *area* and $W \in [W_a, W_b]$. Note that D is constant for a given discrete solution S.

Step 3) Linear program P_W is solved by CPLEX to obtain its optimal solution for a given S and W. This solution has minimal cost among the continuous solutions in $AD(S)$ whose width is less than or equal to W.

In practice, let W increase from W_a to W_b by a small step size, and then use CPLEX to solve P_W for each W value, obtaining a set $C'(S)$ instead of $C(S)$. The $C'(S)$ produced by LPM2 is illustrated in Fig. 3, where each point on the line represents a continuous solution in $C'(S)$.

Each discrete solution S may produce a different line in the objective function space. In fact, the Pareto front of the EDRLP takes the form of multiple disconnected segments, such that each segment corresponds to a unique machine sequence and solutions within a segment represent changes in the absolute position of the machines in real-valued space. Hence, the EDRLP is a truly disconnected multiobjective optimization problem [20], as its Pareto front is in the form of multiple disconnected segments. This is challenging to existing multiobjective optimization algorithms because these algorithms try to join the segments when in fact that is not possible.

In the next section we employ MTS to find a set of non-dominated discrete solutions, where the corresponding set $C'(.)$ may be generated for each of those discrete solutions.

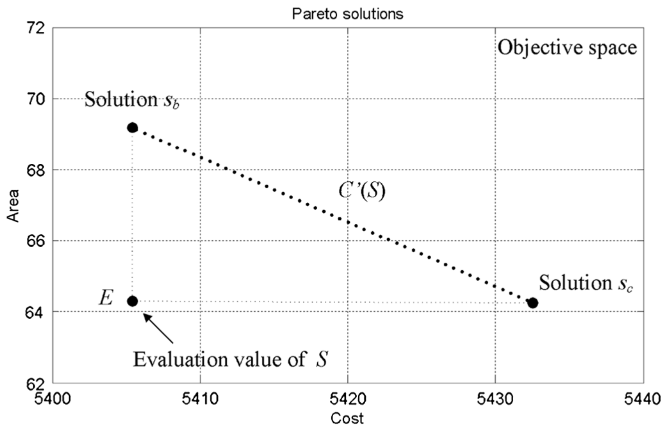

Fig. 3. Non-dominated continuous solutions of a discrete solution S, comprising points in set $C'(s)$. The discrete solution S is evaluated at E.

V. Multiobjective TS With Linear Programming

Tabu search has proven to be effective for combinatorial optimization problems (c.f., [21]), and has been applied to job shop scheduling [22], vehicle routing problems [23] and facility layout problems [10]. As a metaheuristic, TS starts with an initial (generally random) solution, x. In each iteration the current solution is updated to one of its neighboring solutions via a *move*, which is a mechanism that switches from one solution to another. All possible moves from current solution x comprise the move set, $V(x)$, such that all solutions reachable by one move from x represent x's neighborhood. To avoid search cycles and to lead the search to regions with favorable objective values, some aspect of the search history is stored in a *tabu list*, T, that prohibits moves to certain neighboring solutions. Thus, the best non-tabu move is selected from $V(x)$ and is applied to the current solution, x, resulting in a new solution x'. The current solution x is replaced by the new solution x' (i.e., $x = x'$), and some aspect of the move used to create x' is added to the tabu list, T.

A. Finding Non-Dominated Discrete Solutions Using MTS

In this paper, a MTS is applied to find a set of non-dominated discrete solutions. In this MTS, all *solutions* refer to discrete solutions and non-dominated discrete solutions are stored in the set $Archive$. The set of all non-dominated continuous solutions is constructed by

$$NS = \bigcup_{S \in Archive} C'(S).$$

The set of final Pareto solutions of EDRLP is generated by

$$PS = \{s | s \in NS \quad \text{and} \quad \nexists s' \in NS : s' \succ s\}. \tag{24}$$

The steps of MTS are as follows.

Step 1) Initialization. A feasible initial solution is randomly produced as the current solution x. Let $T = \varnothing$, $Archive = \varnothing$ and $iter = 0$. Initialize the length of the tabu list as a random integer in the range of $[minTabu, maxTabu]$.

Step 2) Randomly select a single objective as the active one from the two objectives in (1).

Step 3) A move is defined as the swap of locations of any two machines. Construct the move set $V(x)$ of the current solution x. Moves in the set $V(x)$ are applied to the current solution x to form its neighborhood set $N(x)$. Calculate the objective function value of each neighbor in set $N(x)$.

Step 4) Candidate solutions set $D(x)$ consists of all non-tabu solutions in $N(x)$ and those that are in $N(x)$ and T but dominate any solution in set $Archive$ (an aspiration criterion).

Step 5) Select the best non-tabu solution x' from set $D(x)$ to replace the current solution, i.e., $x = x'$. Update tabu list T and $Archive$ using x'.

Step 6) A local search insertion is used to change the location of a machine from one row to the other. Construct the insertion set $E(x)$ of x, and then select the best solution x'' from $E(x)$. If the objective function value of x'' is better than that of x, then let $x = x''$; otherwise, x'' is discarded. Update set $Archive$ using x''.

Step 7) A restart is used to ensure search diversity. If set $Archive$ is not updated for $reintMax$ iterations, then one of solutions in $Archive$ is randomly selected to replace the current solution x, and the tabu list $T = \varnothing$. Then the search restarts from this new current solution.

Step 8) Let $iter = iter + 1$. If $iter$ is a multiple of 20, update the length of the tabu list as a random integer in the range of $[minTabu, maxTabu]$. If the number of iterations for which $Archive$ set is not updated is less than $maxIter$, return to Step 2; otherwise, the algorithm stops.

1) Evaluation of a Discrete Solution: To find a set of non-dominated discrete solutions, it is necessary to evaluate the quality of a discrete solution. A discrete solution S may correspond to an infinite number of non-dominated continuous solutions in $C(S)$. So, it is required to determine which solution from $C(S)$ is the most appropriate to evaluate the discrete solution S. The two boundary points s_c and s_b are chosen to evaluate S (i.e., S is evaluated by the point E in the objective space, as shown in Fig. 3). This means that S is evaluated by the area objective value of s_c and the cost objective value of s_b.

As discussed in Section IV, s_a and s_c have the same area. Since the area of s_a is easily calculated by using the minimal clearances between adjacent machines, the area of s_a is computed in the evaluation process, instead of s_c. The cost objective value of s_b can be calculated by LPM1. This evaluation requires running CPLEX (in LPM1) for each discrete solution, which is time consuming. To reduce the computational time for solution evaluation, this method is only used to evaluate the best solutions in $N(x)$ and $E(x)$, namely, x' and x'', before they are used to update the set $Archive$.

Each candidate solution in $N(x)$ and $E(x)$ is evaluated by an approximate evaluation method to save computational time. S can be approximately evaluated by the area and cost objective values of s_a. The area and cost of s_a can be calculated using the minimal clearances a_{ij} and parameters f_{ij}.

2) Neighborhood, Moves and Tabu List: In each iteration of MTS, when the cost objective function is selected as the active one, the number of solutions reached by all moves from a current solution x is $n(n-1)/2$, where n is the number of machines. When the area objective function is chosen as the active one, the neighborhood size can be reduced by excluding those moves that cannot improve the area objective.

We observe that swapping any two machines belonging to the shorter row cannot reduce the objective function value of area. In the approximate evaluation method, a discrete solution is evaluated by s_a, which is constructed by separating any two machines by their minimal clearance. As shown in Fig. 2, the length of row 2 is shorter than that of row 1. In this case, swapping any two machines in row 2 cannot reduce the objective value of area. If this swap leads to increasing the length of row 2, making the length of row 2 greater than that of row 1, then this swap will increase the total area; if this swap results in the length of row 2 remaining shorter than that of row 1, then the area is unchanged. The layout area is determined by the longer row, which is called the *critical row* and the machines on the critical row are termed *critical machines*. For example, in solution S, row 1 is the critical row and machines 8, 7, 2, 3, and 1 are critical machines. The move set $V(x)$ is constructed using only moves where there is at least one critical machine.

The move used to create x' is added to the tabu list, T, which contains the most-recently swapped machine pairs. If T is full, the earliest move in the list is removed and the current move is appended.

3) Insertion Operation: The *insertion* operation is a local search to change the assignment of a machine from one row to the other. Each machine in row 1 (row 2) is considered to be inserted into any location in row 2 (row 1). For an EDRLP with n machines, if a discrete solution has p machines in row 1 and q machines in row 2, $p+q=n$, then there are $p(q+1)+(p+1)q = 2pq+n$ candidate solutions in the insertion set $E(x)$.

In each iteration of MTS, when the objective function of cost is selected as the active one, then the number of candidate solutions in $E(x)$ is $2pq+n$. When the area objective function is selected, the size of set $E(x)$ can be reduced by ignoring those candidate solutions that cannot decrease the objective value of area. In this case, inserting a machine into the critical row will result in increasing area, so that machines belonging to the shorter row are forbidden to be inserted into the critical row. Assume row 1 is the critical one, then the number of candidate solutions in $E(x)$ is $p(q+1) < 2pq+n$.

VI. Experimental Results and Analysis

Our approach is applied to several typical industrial problem settings and compared with an exact approach (CPLEX) and a popular multiobjective genetic algorithm (NSGA-II).

A. Test Problems

Problem set P_1 contains small-scale problems with 10 machines, and is used to compare MTS-LP with an exact approach to determine if MTS-LP is able to achieve the Pareto optimal front. Problem sets P_2 and P_3 consist of 20 and 30 machines, respectively, which cannot be solved by the exact approach in an acceptable time. They are used to test the ability of MTS-LP to handle large-sized problem instances.

Within each problem set three aisle widths between the two rows are investigated (i.e., $c_1 = 0$, $c_2 = (\max_{i\in I}\{d_i\})/2$ and $c_3 = \max_{i\in I}\{d_i\}$). By combining three problem data sets with three widths a total of nine instances are produced, namely, $P_1(c_1)$, $P_1(c_2)$, $P_1(c_3)$, $P_2(c_1)$, $P_2(c_2)$, $P_2(c_3)$, $P_3(c_1)$, $P_3(c_2)$, and $P_3(c_3)$. For each problem instance, the width, w_i, and depth, d_i, of each machine are chosen as $\sim$ unif[0.5, 2.5]. The minimum required clearance between machines, a_{ij}, is $\sim$ unif[0.25, 1.5]. Unit flow costs, f_{ij}, are generated by considering 8–10 product types, where each type may have 20–50 units of production. Each product type is assumed to visit between 25% and 75% of the machines, thus determining a route for each product type. Each f_{ij} value is calculated as the sum of products whose routes include machine i immediately preceding machine j. All test problems are available in the accompanying spreadsheets and represent what might be typically found in a semiconductor manufacturing setting.

TABLE II
Algorithm Parameters

Problem Instance	maxIter	reintMax	minTabu	maxTabu
$P_1(c_1)$	750	50	2	4
$P_1(c_2)$	250	50	2	4
$P_1(c_3)$	250	50	2	4
$P_2(c_1)$	1000	500	5	8
$P_2(c_2)$	1000	500	5	8
$P_2(c_3)$	1000	500	5	8
$P_3(c_1)$	1250	500	7	11
$P_3(c_2)$	1000	500	7	11
$P_3(c_3)$	1000	500	7	11

B. Algorithm Parameters

The parameters of MTS-LP consist of those of the TS (shown in Table II) and the step size of area A in LPM2. Note that performance is not sensitive to the TS parameter settings and that these values were chosen after brief experimentation. The step size of area in LPM2 depends on the number of continuous solutions in PS (see (24)) desired, such that a smaller step size yields more Pareto solutions. For problem instances with ten machines, the step size is chosen to be 0.1, and for instances with 20 and 30 machines, the size is 0.3 and 0.5, respectively. For all problem instances, the maximum size of the $Archive$ set is 30.

C. Comparative Algorithms

1) Exact Approach: CPLEX 12.0 is used to solve the formulation in Section III, where the two objectives are combined linearly to form a single objective as shown in (25)

$$\text{Minimize } \alpha \sum_{i\in I^1}\sum_{j\in I^2}(f_{ij}+f_{ji})\left(v_{ij}^{+}+v_{ij}^{-}+c(1-q_{ij})\right) + (1-\alpha)A. \quad (25)$$

To mimic the continuous nature of α, let α increase from 0 to 1 by a small step size, using CPLEX to solve the formulation for each α value. By this means, CPLEX is able to find multiple Pareto solutions but not all of them because of the limited number of α values used. For problem instances with ten machines, CPLEX is able to find the optimal solution for each given α value quickly (within 90–150s). In this case, the step size of α is chosen as 0.02. For instances with 20 and 30 machines, CPLEX cannot obtain optimal solutions within an acceptable

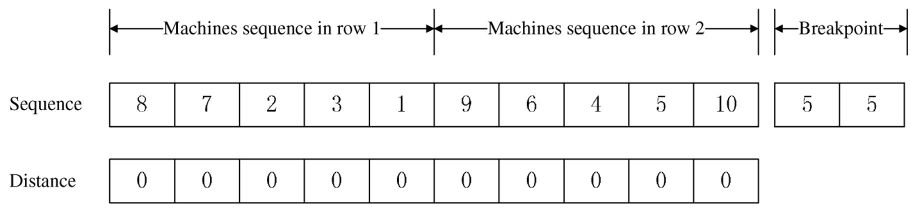

Fig. 4. Representation of a solution in the NSGA-II for EDRLP.

time, and the step size is assigned to 0.1. Here, the run time of CPLEX is restricted to 1 hour for each α value.

2) NSGA-II: Because the EDRLP defined in this paper is a newly modeled and solved problem, there are no benchmark solution methodologies. We have selected a well known multi-objective optimization algorithm, NSGA-II [24] with which to compare. Reference [25] have demonstrated the use of NSGA-II to solve a block layout problem in a flexible bay structure [26] where the number of bays and the sequence of departments in each bay must be specified (although the exact location for each department is not identified). In this paper, we extend the NSGA-II algorithm of [25] to solve the EDRLP.

a) Solution representation: In [25], a solution of the block layout problem denotes the sequence of departments in each bay, and is represented by two integer vectors: the first one is a permutation of all departments and the second one (breakpoints) identifies the number of departments in each bay. We may use the solution representation of [25] directly to express the machine sequences in the EDRLP. However, to represent exact machine locations, we devise a real-valued vector, the ith gene (element) of which denotes the additional distance between the left side of machine i and the right side of the machine immediately to its left (let this be machine j). This distance is in addition to the minimum required separation, a_{ij}.

For example, the coding of solution s_a shown in Fig. 2 is given in Fig. 4, where the second integer vector [5 5] indicates the existence of five machines in row 1 and 5 in row 2. The sequence of machines in each row is identified by the first integer vector. Each gene of the real-valued distance vector equals zero because each pair of adjacent machines in s_a is separated by their minimal clearance (i.e., no additional clearance is observed).

b) Genetic operators: In NSGA-II, the integer and real-valued codings perform different genetic operations (i.e., crossover and mutation operators), respectively. For the integer coding, we use the genetic operators in [25], which involve the generic uniform-based crossover and random mutation. For the real-valued one, the simulated binary crossover (SBX) and the real-parameter mutation in [24] are used. Other operators, such as the selection operator, non-dominated sorting, and archive set updating, are as standard in NSGA-II.

c) Initialization: For each individual in the initial population, its first integer vector is initialized as a random permutation of all machines and each gene of its real-parameter vector is assigned to a random real number in $[0, \sum_{i \in I} w_i/2]$. The first gene of its second integer vector is set to a random integer in $[m/3, 2m/3]$ (where m is the number of machines), and let the second gene equal m minus the first one. We do this to restrict the difference between the number of machines in the two rows since this difference is usually not very large for a Pareto solution.

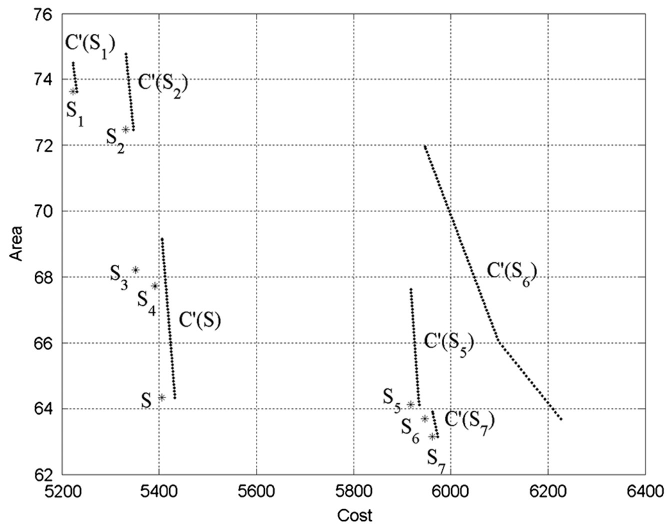

Fig. 5. Set NS produced using LPM2 for $P_1(c_2)$.

d) Parameters setting: The NSGA-II implementation of [24] uses a population size of 100, a crossover probability of 0.8, and a mutation probability of $1/n$, where n is the length of a solution coding. We also adopt those settings when using NSGA-II to solve the EDRLP, except that n denotes the number of machines instead of the length of coding.

While [24] employed 250 generations for NSGA-II, we increased the number of generations to 20,000 for problem instances with 10 machines in an effort to improve convergence. Problem instances with 20 and 30 machines were allowed to run for 50,000 and 70,000 generations, respectively. We observed that larger numbers of generations did not improve the quality of solutions for our test problem instances.

D. Experimental Results

The MTS-LP and NSGA-II algorithms were coded in C, and all code (including CPLEX 12.0) was executed on an HP 8100 Elite desktop PC with a quad-core Intel i7-860 processor running Ubuntu Linux 10.10 in 64-bit mode. Five independent runs of MTS-LP and NSGA-II (using a different random starting point for each run) were performed for each of the nine problem instances.

1) Small Size Problem Instances: For problem $P_1(c_2)$, each of the five runs of MTS was able to find the same non-dominated discrete solutions, indicating that the performance of MTS is stable for small size instances. For each discrete solution found by MTS, a set of non-dominated continuous solutions can be produced using LPM2. The set NS of all non-dominated continuous solutions for $P_1(c_2)$ is illustrated in Fig. 5, in which the continuous solutions are denoted by $\cdot$. Since the five runs of MTS obtain the same discrete solutions, the non-dominated continuous solutions are also the same. Eight sets of non-dominated

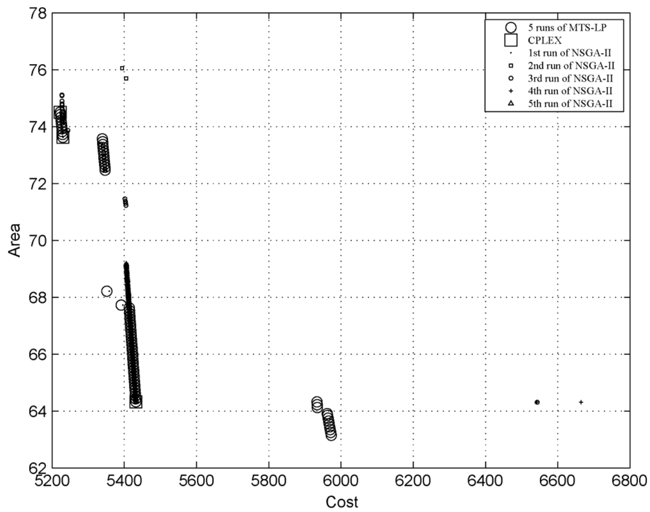

Fig. 6. Pareto solutions found by MTS-LP, NSGAII, and CPLEX for $P_1(c_2)$.

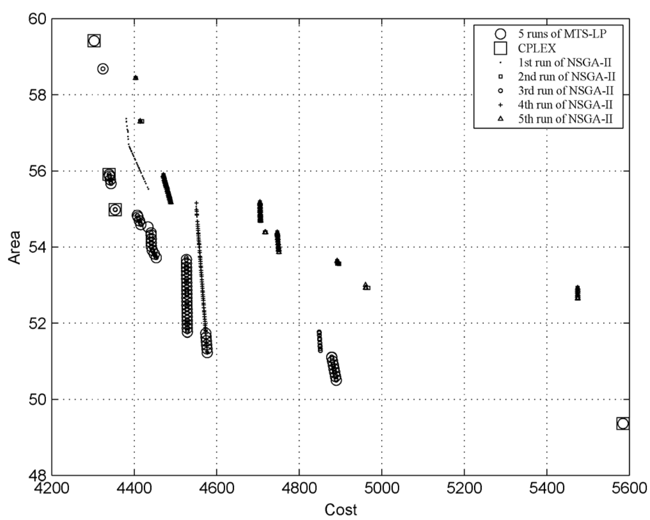

Fig. 7. Pareto solutions found by MTS-LP, NSGAII, and CPLEX for $P_1(c_1)$.

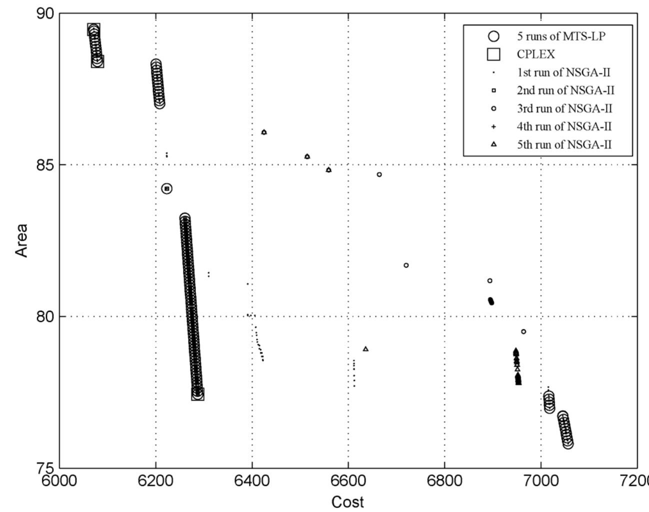

Fig. 8. Pareto solutions found by MTS-LP, NSGAII, and CPLEX for $P_1(c_3)$.

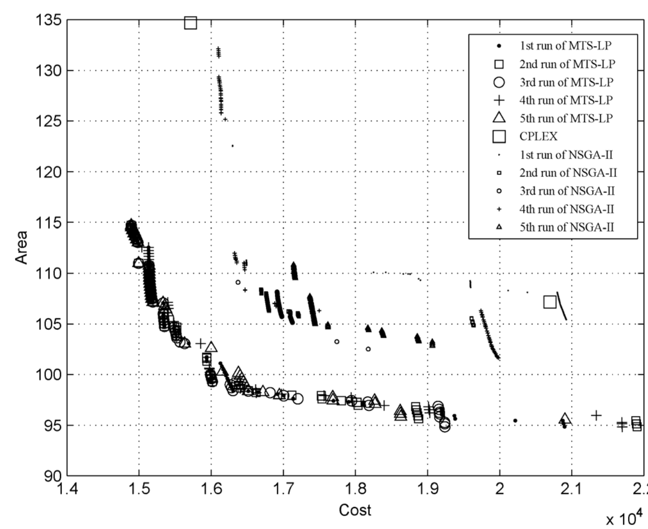

Fig. 9. Pareto solutions found by MTS-LP, NSGAII, and CPLEX for $P_2(c_2)$.

continuous solutions are generated, each of which corresponds to a discrete solution denoted by $*$ in the figure. The solution S in the figure is used as an example in Section IV, and its set $C'(S)$ is also illustrated in Fig. 3.

The final Pareto solution set PS of $P_1(c_2)$ is generated by (24), represented by large $\circ$ in Fig. 6. It can be observed that the Pareto set consists of several disconnected segments, each of which corresponds to a discrete solution.

For problems $P_1(c_1)$ and $P_1(c_3)$ MTS can still find the same discrete solutions in each of its five runs. However, since there are more non-dominated discrete solutions for $P_1(c_1)$ (in the case of $c = 0$), it is more difficult to find them all. Therefore, parameter *maxIter* is given a larger value for $P_1(c_1)$. The final Pareto solutions of $P_1(c_1)$ and $P_1(c_3)$ are illustrated by a large $\circ$ in Figs. 7 and 8, respectively.

Pareto solutions found by the exact approach (CPLEX) and NSGA-II are also given in Figs. 6–8, respectively. It can be seen that for each of the three instances CPLEX is able to obtain several Pareto solutions, which are all located at the boundaries of the set of non-dominated continuous solutions $C'(.)$. Since these are the same as for MTS-LP, we can conclude that MTS-LP is able to find the true Pareto front for small size problems. Moreover, MTS-LP can find Pareto solutions that are distributed along the Pareto front, while CPLEX can only find the boundary solutions (that is, s_c and s_b). From Figs. 6–8, we can see that the solutions obtained by NSGA-II are worse than those of MTS-LP. Each run of NSGA-II only finds a part of the Pareto front obtained by MTS-LP, or cannot find the front at all. The reason may be twofold: 1) MTS is more effective and stable than NSGA-II for searching the non-dominated machine sequences and 2) for a given machines sequence, LPM2 can find all its Pareto optimal solutions, while NSGA-II cannot guarantee finding the optimal ones since it is a heuristic. In addition, both NSGA-II and CPLEX can only obtain a limited number of Pareto solutions, while MTS-LP can produce as many Pareto solutions as desired by reducing the step size of area (α).

Table III provides the mean number of iterations of MTS, and the mean, maximal and minimal runtime over the five runs of MTS, the mean time to produce the set PS by LPM2, and the total run time of MTS-LP. The mean run time over the five runs of NSGA-II and the run time of CPLEX are also presented.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

TABLE III
COMPARISON OF RUNTIME FOR THE INSTANCES WITH TEN MACHINES

Instance	MTS-LP						CPLEX time (s)	NSGA-II	
	TS mean iters	TS run time (s)			PS mean time (s)	Total time (s)		Mean iters	Mean time (s)
		Max	Min	Mean					
$P_1(c_1)$	22100	290.59	20.93	182.18	5.11	187.29	7050.21	20000	41.20
$P_1(c_2)$	2470	44.64	35.98	41.81	5.46	47.27	6490.49	20000	42.40
$P_1(c_3)$	2201	44.29	17.76	35.43	3.92	39.35	5034.67	20000	43.80

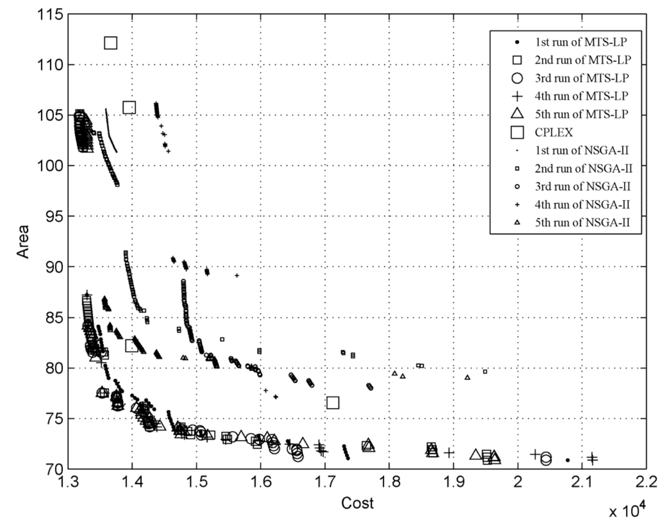

Fig. 10. Pareto solutions found by MTS-LP, NSGAII, and CPLEX for $P_2(c_1)$.

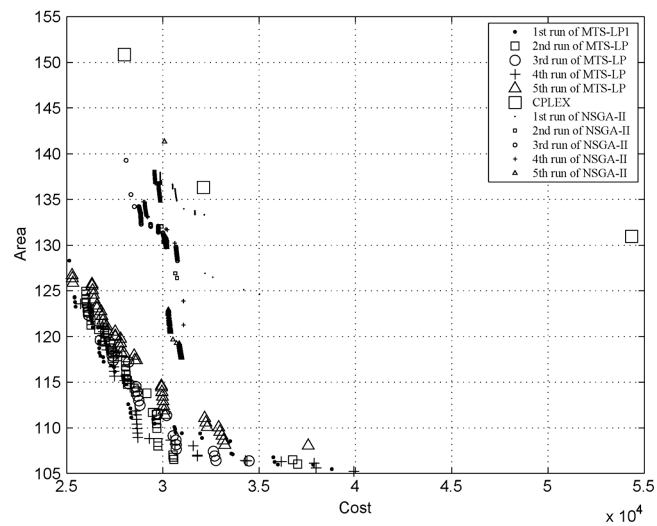

Fig. 12. Pareto solutions found by MTS-LP, NSGAII, and CPLEX for $P_3(c_1)$.

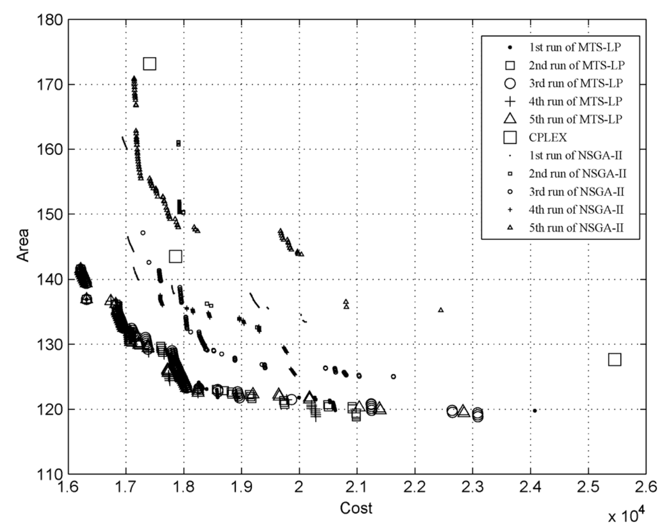

Fig. 11. Pareto solutions found by MTS-LP, NSGAII, and CPLEX for $P_2(c_3)$.

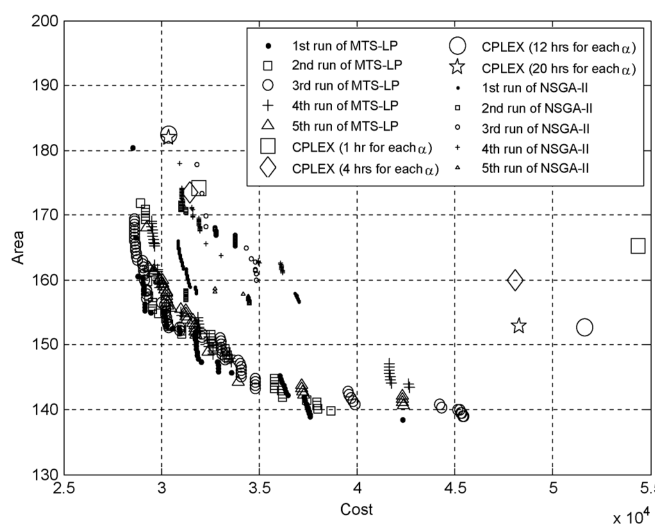

Fig. 13. Pareto solutions found by MTS-LP, NSGAII, and CPLEX for $P_3(c_2)$.

From this table, it can be seen that MTS-LP can obtain a set of Pareto solutions for $P_1(c_2)$ and $P_1(c_3)$ within 1 min and takes about 3 min for $P_1(c_1)$, while CPLEX takes about 2 hours and only obtains several Pareto solutions. The run time of NSGA-II is similar to that of MTS-LP for instances $P_1(c_2)$ and $P_1(c_3)$, and shorter than MTS-LP for $P_1(c_1)$.

2) Large Size Problem Instances: The final Pareto solutions found by MTS-LP, NSGA-II, and CPLEX for $P_2(c_2)$, $P_2(c_1)$, and $P_2(c_3)$ are illustrated in Figs. 9–11, respectively. It can be seen that MTS-LP is able to obtain much better Pareto solutions than NSGA-II and CPLEX. The width between rows, c, does not appear to have much influence on the performance of MTS-LP. The quality of individual solutions found by NSGA-II and CPLEX is similar, although NSGA-II is able to find more Pareto solutions than CPLEX. It should be noted that the performance of NSGA-II is not stable, as each of its runs produces different Pareto solutions and they are all worse than those found by MTS-LP.

A comparison of run times for MTS-LP, NSGA-II, and CPLEX is given in Table IV. The run time of CPLEX for each problem instance is 11 hours, while the run time of MTS-LP is about 7–10 min. NSGA-II requires a shorter run

TABLE IV
COMPARISON OF RUNTIME FOR INSTANCES WITH 20 MACHINES

Instance	MTS-LP						CPLEX time (hr)	NSGA-II	
	TS mean iters	TS run time (s)			PS mean time (s)	Total time (s)		Mean iters	Mean time (s)
		Max	Min	Mean					
$P_2(c_1)$	21541	605.80	106.24	438.14	19.01	457.15	11	50000	160.20
$P_2(c_2)$	20471	621.57	184.89	423.91	21.82	445.73	11	50000	140.40
$P_2(c_3)$	28295	626.68	529.90	585.18	24.14	609.32	11	50000	146.20

TABLE V
COMPARISON OF RUNTIME FOR INSTANCES WITH 30 MACHINES

Instance	MTS-LP						CPLEX time (hr)	NSGA-II	
	TS mean iters	TS run time (s)			PS mean time (s)	Total time (s)		Mean iters	Mean time (s)
		Max	Min	Mean					
$P_3(c_1)$	33763	2759.64	281.47	1549.62	39.61	1589.23	11	70000	343.80
$P_3(c_2)$	14774	1296.33	247.33	721.22	49.63	770.85	11	70000	348.20
$P_3(c_3)$	13168	1060.70	187.29	625.79	59.40	685.19	11	70000	354.40

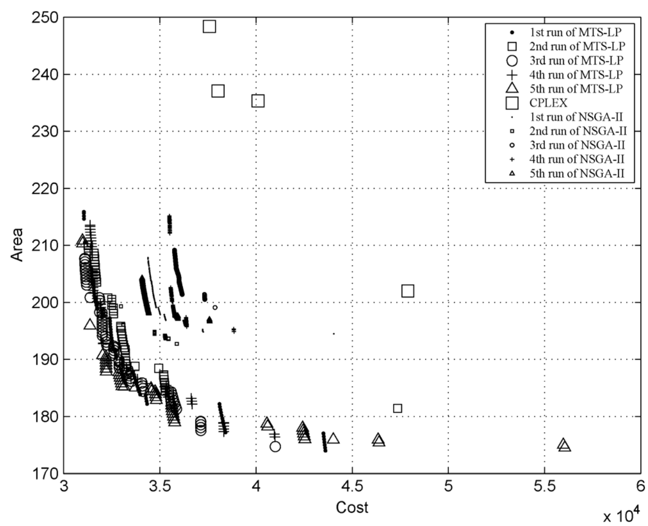

Fig. 14. Pareto solutions found by MTS-LP, NSGAII, and CPLEX for $P_3(c_3)$.

time (2–3 min) because it is a heuristic that does not involve any exact approach.

Pareto solutions obtained by MTS-LP, NSGA-II, and CPLEX for $P_3(c_1)$, $P_3(c_2)$, and $P_3(c_3)$ are shown in Figs. 12– 14, respectively. A comparison of run time of MTS-LP, NSGA-II, and CPLEX for the three instances is given in Table V. The results here echo those of the preceding problem set.

To investigate the Pareto solutions found by CPLEX when run longer, the time for $P_3(c_2)$ is increased to 4 hours for each α value (for a total runtime of 44 hours). Resulting Pareto solutions are represented by large $\diamond$ in Fig. 13, where it is clear that CPLEX still cannot obtain satisfactory Pareto solutions. In fact, a small test for α values of 0 and 0.7 reveals that if the runtime were increased to 12 hours (represented by large circles) and 20 hours (large stars) for each α, CPLEX is still unable to obtain Pareto solutions of a quality comparable to those found by MTS-LP.

VII. CONCLUSION

In this paper, a common problem in the microelectronics manufacturing industry is formulated and solved for the first time. This problem includes both discrete and continuous decision variables, and its Pareto front consists of several disconnected segments, each of which is a set of continuous Pareto solutions.

Experimental results show that MTS-LP greatly improves upon NSGA-II and an exact approach for problems of more than trivial size. The MTS-LP is not sensitive to search parameter values or to starting solutions. It runs quickly and covers the breadth of the Pareto front well. The solutions allow a decision maker to choose amongst superior layout possibilities, depending on the preference between a small footprint versus less operational cost in the form of material handling savings. Because these costs are dissimilar—the first a onetime cost while the second an ongoing cost—combining them into a single objective would be difficult from a decision making perspective.

Future work includes examining this problem for multiple manufacturing periods and for more than two rows of machines. Another aspect is to consider relayout where there are existing machines and utilities and the cost of rearrangement must be considered.

ACKNOWLEDGMENT

The authors wish to thank the Editor, Associate Editor, and the anonymous referees for their valuable feedback and suggestions.

REFERENCES

[1] R. D. Meller and K. Y. Gau, "The facility layout problem: Recent and emerging trends and perspectives," *J. Manuf. Syst.*, vol. 15, no. 5, pp. 351–366, 1996.

[2] A. Drira, H. Pierreval, and S. Hajri-Gabouj, "Facility layout problems: A survey," *Annu. Rev. Control*, vol. 31, no. 3, pp. 255–267, 2007.

[3] J. Turley, *The Essential Guide to Semiconductors*. Englewood Cliffs, NJ, USA: Prentice-Hall, 2002.

[4] J. Chung and J. M. A. Tanchoco, "The double row layout problem," *Int. J. Prod. Res.*, vol. 48, no. 3, pp. 709–727, 2010.

[5] S. P. Singh and R. R. K. Sharma, "A review of different approaches to the facility layout problems," *Int. J. Adv. Manuf. Technol.*, vol. 30, no. 5–6, pp. 425–433, 2006.

[6] G. Moslemipour, T. S. Lee, and D. Rilling, "A review of intelligent approaches for designing dynamic and robust layouts in flexible manufacturing systems," *Int. J. Adv. Manuf. Technol.*, vol. 60, no. 1–4, pp. 11–27, 2012.

[7] A. R. S. Amaral, "On the exact solution of a facility layout problem," *Eur. J. Oper. Res.*, vol. 173, no. 2, pp. 508–518, 2006.

[8] F. Ozcelik, "A hybrid genetic algorithm for the single row layout problem," *Int. J. Prod. Res.*, vol. 50, no. 20, pp. 5872–5886, 2012.

[9] R. Kothari and D. Ghosh, "Tabu search for the single row facility layout problem using exhaustive 2-opt and insertion neighborhoods," *Eur. J. Oper. Res.*, vol. 224, pp. 93–100, 2013.

[10] H. Samarghandi and K. Eshghi, "An efficient tabu algorithm for the single row facility layout problem," *Eur. J. Oper. Res.*, vol. 205, no. 1, pp. 98–105, 2010.

[11] C. Ou-Yang and A. Utamima, "Hybrid estimation of distribution algorithm for solving single row facility layout problem," *Comput. Ind. Eng.*, vol. 66, no. 1, pp. 95–103, 2013.

[12] S. S. Heragu, "Recent models and techniques for solving the layout problem," *Eur. J. Oper. Res.*, vol. 57, no. 2, pp. 136–144, 1992.

[13] B. A. Peters and T. Yang, "Integrated facility layout and material handling system design in semi-conductor fabrication facilities," *IEEE Trans. Semicond. Manuf.*, vol. 10, no. 3, pp. 360–369, 1997.

[14] K. S. N. Ripon, K. N. Khan, K. Glette, M. Hovin, and J. Torresen, "Using Pareto-optimality for solving multi-objective unequal area facility layout problem," in *Proc. 13th Annu. Conf. Genetic Evol. Comput.*, 2011, pp. 681–688.

[15] A. R. S. Amaral, "The corridor allocation problem," *Comput. Oper. Res.*, vol. 39, pp. 3325–3330, 2012.

[16] H. Ahonen, A. G. de Alvarenga, and A. R. S. Amaral, "Simulated annealing and tabu search approaches for the Corridor Allocation Problem," *Eur. J. Oper. Res.*, vol. 232, no. 1, pp. 221–233, 2014.

[17] Z. Q. Zhang and C. C. Murray, "A corrected formulation for the double row layout problem," *Int. J. Prod. Res.*, vol. 50, no. 15, pp. 4220–4223, 2012.

[18] A. R. S. Amaral, "Optimal solutions for the double row layout problem," *Opt. Lett.*, vol. 7, no. 2, pp. 407–413, 2013.

[19] C. C. Murray, X. Q. Zuo, and A. E. Smith, "An extended double row layout problem," in *Proc. 12th Int. Mater. Handling Res. Colloquium*, Gardanne, France, 2012, to be published.

[20] C. K. Chow and S. Y. Yuen, "A multiobjective evolutionary algorithm that diversifies population by its density," *IEEE Trans. Evol. Comput.*, vol. 16, no. 2, pp. 149–172, Apr. 2012.

[21] S. Kulturel-Konak, A. E. Smith, and D. A. Norman, "Multi-objective tabu search using a multinomial probability mass function," *Eur. J. Oper. Res.*, vol. 169, no. 3, pp. 918–931, 2006.

[22] F. Pezzella and E. Merelli, "A tabu search method guided by shifting bottleneck for the job shop scheduling problem," *Eur. J. Oper. Res.*, vol. 120, no. 2, pp. 297–310, 2000.

[23] E. E. Zachariadis, C. D. Tarantilis, and C. T. Kiranoudis, "A guided tabu search for the vehicle routing problem with two-dimensional loading constraints," *Eur. J. Oper. Res.*, vol. 195, no. 3, pp. 729–743, 2009.

[24] K. Deb, S. Agrawal, A. Pratap, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Trans. Evol. Comput.*, vol. 6, no. 2, pp. 182–197, Apr. 2002.

[25] G. Aiello, M. Enea, and G. Galante, "A multi-objective approach to facility layout problem by genetic search algorithm and Electre method," *Robot. Comput.-Integr. Manuf.*, vol. 22, pp. 447–455, 2006.

[26] D. M. Tate and A. E. Smith, "Unequal-area facility layout by genetic search," *IIE Trans.*, vol. 27, pp. 465–472, 1995.

Xingquan Zuo (M'09) received the Ph.D. degree in control theory and control engineering from the Harbin Institute of Technology, Harbin, China, in 2004.

He is currently an Associate Professor with the Computer School, Beijing University of Posts and Telecommunications. From 2004 to 2006, he was a Postdoctoral Research Fellow with the Automation Department, Tsinghua University. From 2012 to 2013, he was a Visiting Scholar in the Industrial and System Engineering Department, Auburn University, USA. His research interests are in system optimization and scheduling, evolutionary computation, and intelligent transportation system. He has published over 50 research papers in journals and conferences, two books, and several book chapters.

Dr. Zuo is a member of the IEEE Computational Intelligence Society, Committee Member of the Natural Computation Society of Chinese Association for Artificial Intelligence, and Senior Member of the Chinese Association for Artificial Intelligence.

Chase C. Murray received the B.S. and M.Eng. degrees in industrial engineering from Texas A&M University, College Station, TX, USA, in 1998 and 2000, respectively, and the Ph.D. degree in industrial and systems engineering from the University at Buffalo, Buffalo, NY, USA, in 2010.

He is an Assistant Professor of Industrial and Systems Engineering at Auburn University, Auburn, AL, USA. His research interests are in the area of operations research, with a particular focus on military applications. His other areas of interest include vehicle routing and scheduling, as well as facility layout. He previously worked as an Industrial Engineer in the semiconductor industry as an employee of Intel and Dallas Semiconductor.

Dr. Murray is a Member of the Institute for Operations Research and the Management Sciences (INFORMS), the Institute of Industrial Engineers (IIE), the Production and Operations Management Society (POMS), and the Society for Industrial and Applied Mathematics (SIAM).

Alice E. Smith received the B.S.C.E. degree from Rice University, Houston, TX, USA, the MBA degree from Saint Louis University, St. Louis, MO, USA, and the Ph.D. degree from Missouri University of Science and Technology, Rolla, MO, USA.

She is the W. Allen and Martha Reed Professor of the Industrial and Systems Engineering Department, Auburn University, Auburn, AL, USA, where she served as Department Chair from 1999 to 2011. She also has a joint appointment with the Department of Computer Science and Software Engineering. Previously, she was on the faculty of the Department of Industrial Engineering, University of Pittsburgh from 1991 to 1999, which she joined after industrial experience with Southwestern Bell Corporation. For accomplishments in research, education and service she was named the H. Allen and Martha Reed Endowed Professor in 2012. She holds one U.S. patent and several international patents and has authored more than 200 publications which have garnered over 1900 citations (ISI Web of Science).

Dr. Smith was awarded the IIE Albert G. Holzman Distinguished Educator Award in 2012 and the INFORMS WORMS Award for the Advancement of Women in OR/MS in 2009. She was named the Philpott-WestPoint Stevens Distinguished Professor in 2001, received the Senior Research Award of the College of Engineering at Auburn University in 2001, and the University of Pittsburgh School of Engineering Board of Visitors Faculty Award for Research and Scholarly Activity in 1996. She has served as Chair of the Council of Industrial Engineering Academic Department Heads and as President of the INFORMS Association of Chairs of Operations Research Departments. She won the E. L. Grant Best Paper Awards in 1999 and in 2006, and the William A. J. Golomski Best Paper Award in 2002. Several of her papers are among the most highly cited in their respective journals including the second most cited paper of IEEE TRANSACTIONS ON RELIABILITY and the second most cited paper of *Reliability Engineering and System Safety*. She is an Area Editor of both the *INFORMS Journal on Computing and Computers and Operations Research* and an Associate Editor of IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION. She has been a principal investigator on over $6 million of sponsored research. Her research in analysis, modeling and optimization of complex systems has been funded by NASA, the U.S. Department of Defense, the Missile Defense Agency, NIST, the U.S. Department of Transportation, Lockheed Martin, Adtranz (now Bombardier Transportation), the Ben Franklin Technology Center of Western Pennsylvania, and the U.S. National Science Foundation, from which she has been awarded 16 grants including a CAREER grant in 1995 and an ADVANCE Leadership grant in 2001. International research collaborations have been sponsored by the federal governments of Japan, Turkey, United Kingdom, The Netherlands, Egypt, South Korea, Iraq, China, Algeria, and the U.S., and by the Institute of International Education. In 2013, she was a Fulbright Senior Scholar at Bilkent University, Turkey. She is a Fellow of the Institute of Industrial Engineers, a Senior Member of the Institute of Electrical and Electronics Engineers (IEEE) and of the Society of Women Engineers, a Member of Tau Beta Pi, and the Institute for Operations Research and Management Science, and a Registered Professional Engineer in Alabama and Pennsylvania. She was elected to serve on the Administrative Committee of the IEEE Computational Intelligence Society from 2013-2015 and as IIE Senior Vice President–Publications from 2014-2017.