Computers and Operations Research 108 (2019) 20–32

Contents lists available at ScienceDirect

Computers and Operations Research

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

Double-row facility layout with replicate machines and split flows

Mehmet Gülşen [a,*], Chase C. Murray [b], Alice E. Smith [c]

[a] Industrial Engineering Department, Baskent University, School of Engineering, Eskişehir Yolu 20 km., Bağlica Kampüsü Etimesgut, Ankara 068100, Turkey
[b] Department of Industrial and Systems Engineering, University at Buffalo, 309 Bell Hall, Buffalo, NY, USA, 14260-2050
[c] Distinguished Professor, Department of Industrial & Systems Engineering, Auburn University, 3301 Shelby Center, Auburn, AL, USA, 36849-5346

ARTICLE INFO

Article history:
Received 24 November 2017
Revised 31 January 2019
Accepted 20 March 2019
Available online 28 March 2019

Keywords:
Facility planning and design
Double row layout problem
Flow assignment
Alternative machines

ABSTRACT

This paper presents a new variant of the double row layout problem (DRLP) that features types of capacitated machines. This variant dramatically improves the fidelity with which the DRLP represents the complex production alternatives in many production environments, especially in the microcomputer industry. The problem consists of placing different classes of machines, each with multiple replicates, on either side of a central aisle. This is a common type of layout problem observed in many production and service environments. This work extends the DRLP literature by utilizing machine replicates in the layout. Multiple replicates of the same machine allow the product flow to split into multiple branches, each representing a parallel flow line. Flow splitting can improve machine utilization and reduces the number of machines needed. We also introduce machine sharing, which lets different products use the same machine. This eliminates the need for dedicated assignments of products to machines, which is a common assumption in the current literature. The consideration of re-entrant product flow, whereby each product can visit the same machine multiple times, represents another unique aspect of this research. The problem is formulated as a nonlinear mixed integer program. Due to the computational complexity of this problem a hierarchical optimization approach is developed and evaluated on test problems of various sizes. Computational results show that the proposed method produces consistent and high quality solutions across problem sizes.

1. Introduction

Layout planning in a complex manufacturing environment such as semiconductor and microelectronic production is a challenging task that has a direct influence on plant productivity. Effective space usage and machine layout configuration are critical aspects of total resource management as it directly impacts a company's profitability. This paper is motivated by manufacturers that produce multiple products within the same facility, as is the case in semiconductor and microelectronics manufacturing. Such manufacturers typically utilize multiple essentially-identical machines of the same class or type, such that a product's process flow is defined by the sequence of machine classes (rather than particular machines) that must be visited. Using identical machines connected in various configurations is common in the semiconductor industry (Bureau et al., 2006).

More effective layouts may be constructed by recognizing that some machines may share capacity across multiple product types. Additionally, material handling costs may be reduced by determining product flows via optimization, rather than by assuming these flows to be pre-specified. For some machine-intensive processes, product flow may split into multiple branches (i.e., parallel machining) and then merge back. This problem is considerably more realistic, and complex, than those previously studied in the literature. We focus our attention on facilities that utilize a double-row layout configuration, such that machines are allocated to both sides of a straight aisle or corridor. We allow for the case that some machines within a particular class may have differing capacities (i.e., the number of product units that can be processed during the planning horizon). That is, while they perform the same operation, some machines may perform that operation more quickly. We refer to the individual machines of a particular type as *replicas*, where all replicas of a particular type serve the same function. However, each replica of a machine type may differ according to its size and its production capacity. Machine capacities are fixed, and may reflect expected downtime. Our model explicitly considers multiple product types, each with its own unique process flow.

We term this problem the *double-row layout problem with multiple machine types* (DRLP-MMT). One unique aspect of the DRLP-MMT is that each product may visit the same machining process

* Corresponding author.
E-mail addresses: mgulsen@baskent.edu.tr (M. Gülşen), cmurray3@buffalo.edu (C.C. Murray), smithae@auburn.edu (A.E. Smith).

https://doi.org/10.1016/j.cor.2019.03.009
0305-0548/

more than once (i.e., re-entrant flow is allowed). Additionally, products of the same type are not restricted to visiting the same sequence of individual machines (i.e., flow "splitting" is allowed). Furthermore, we do not restrict each machine to be dedicated to a single product, thus allowing for better utilization of capital equipment (and reflecting the realities of semiconductor manufacturing).

While the single row facility layout problem (SRFLP) concerns the arrangement of facilities on a single row as in Amaral (2006, 2008); Heragu and Kusiak (1991), the DRLP considers two rows of machines, an upper and a lower. The DRLP requires the determination of both the sequence of machines in each row (relative placement) and the actual location of each machine (absolute placement). It has widespread application in industry, including semiconductor manufacturing as studied in Zuo et al. (2016b), because determining the exact location for each machine may result in a layout with less material flow cost than only choosing the sequence separated by standard clearances.

In its original form, the objective of the DRLP is to determine the sequence and position for each machine to minimize material flow cost. However, the production area may also be an important factor, such as in high value manufacturing environments where material flow can be complex and construction costs (e.g., clean room) are approximately $3,500 per square foot or higher (Turley, 2002). Thus, Murray et al. (2012) proposed an extended double row layout problem (EDRLP) with non-zero aisle width and the optimization objectives of both cost and layout area were linearly combined to form a single objective. A commercial IP solver was used to solve this EDRLP, allowing solutions to only small-scale problems. A multi-objective heuristic, suitable for larger problems, was proposed in Zuo et al. (2014).

Although the research on DRLP goes back to Heragu and Kusiak (1988), the first mixed integer formulation of the problem was developed by Chung and Tanchoco (2010). In this paper, the location of each machine is identified by two parameters, an absolute and a relative location which are represented by respective integer and continuous variables in the formulation. Expressing machine location in dual-parameters is a common practice, and it dates back to another work of Heragu and Kusiak (1991) where a binary-continuous variable combination is used in the SRFLP formulation. An alternative to the Chung and Tanchoco (2010) formulation was proposed by Amaral (2013a) with a fewer number of variables and constraints. Over several test cases, Amaral (2013a) showed that his approach performs faster than the formulation proposed by Chung and Tanchoco (2010). The performance of Amaral (2013a) is further improved in a modified formulation of Secchin and Amaral (2018). Recently, Amaral (2018) re-formulates DRLP so that it can be more intuitive in handling qualitative input.

Several papers have appeared to address the "double row layout" problem, although the problem they study is different from that defined by Chung and Tanchoco (2010). The corridor allocation problem (CAP) and parallel row ordering problem (PROP) are based on arrangement of facilities in two rows such that the cost function is minimized. In the CAP formulation by Amaral (2012) there is no gap between the wall and the leftmost facilities. Furthermore, no space is permitted between adjacent facilities. PROP can be considered as an extension of SRFLP, as facilities are restricted to specific rows (Amaral, 2013b). This category of combinatorial problems also includes minimum duplex arrangement problems where n facilities of equal length are assigned to n locations (Amaral, 2011).

Our proposed solution approach combines well-known standard tools (neighborhood search, decomposition, LP) in a unique architecture to solve this very challenging problem. It involves a heuristic that exploits problem structure to decompose the problem into two phases that are combined in a hierarchy. In the upper level (Phase I), machines are assigned to the upper and lower rows. The *relative* ordering of machines in each row is also decided here. For a layout selected in the upper level, the total flow cost is minimized in the lower level (Phase II) by adjusting the distance between adjacent machines and by determining the flow quantities in the network. The objective is to minimize the total flow cost of the layout, which is defined as the product of the flow quantity and distance.

This paper builds upon the existing double-row layout problem (DRLP) literature, previous examples of which are provided by Chung and Tanchoco (2010); Heragu and Kusiak (1988); Zhang and Murray (2012), and Murray et al. (2013). Our work extends the DRLP literature by considering multiple machine classes, each with non-identical machine replicates. Furthermore, this work considers multiple product flows (i.e., simultaneous production of multiple items) in the context of the DRLP. For each product the flow network may include splitting into branches that represent parallel production routes. Allowing flow to split into parallel branches improves machine utilization and can reduce the number of machines required. This is the first paper which considers these important relaxations to the DRLP.

The remainder of this paper is organized as follows. A review of related facility layout literature is provided in Section 2. A formal problem description, including a mixed integer nonlinear programming formulation, is given in Section 3. The proposed heuristic is described in Section 4, followed by a numerical analysis in Section 5. Finally, concluding remarks and opportunities for further research are described in Section 6.

2. Related literature

The research on facilities planning problems is vast, and spans many decades. We restrict our attention to the facilities layout literature that considers "replicas" of machine types, machine capacities, and multiple product flows; such research has grown in popularity in recent years. We organize the review according to problem types along with highlighting of models used for each problem type. We first discuss the relevant literature directly related to the DRLP in terms of problem formulation and solution approach. Secondly, we review DRLPs that are formulated as quadratic assignment problems (QAP). The last part of the review includes works that are not directly related to facility layout problems, but includes solution techniques that are used in our approach.

The DLRP as formulated in this research is based on the mixed integer formulation of Chung and Tanchoco (2010). This original work was modified and expanded by Murray et al. (2013); Zhang and Murray (2012), and Zuo et al. (2016a). In the Chung and Tanchoco (2010) formulation, the integer variables determine the ordering of machines in the upper and lower rows of the layout. The absolute positions of machines are represented by continuous variables. Unlike our approach, the Chung and Tanchoco (2010) formulation does not include machine replicates, re-entrant flows or positive aisle width; it can be solved as a mixed integer program. In terms of solution methodology, our approach is more related to some other works in the facilities layout literature, as discussed below.

Within the context of the DRLP, Castillo and Peters (2004) used a two-stage solution approach for a multi-bay manufacturing facility with replicate machines. There are multiple products and machines with multiple replicas. The problem seeks to minimize the total material handling costs (flow quantity times distance) by assigning machines to bays that are of predefined size and are allocated to predefined positions. Material handling costs are partitioned into inter-bay and intra-bay travel. Within each bay there is a constraint that ensures that the total area of the bay is not exceeded (however there are no "overlap" or "minimum clearance" constraints). This problem does not involve the absolute positioning of machines, only the assignment of machine replicas to bays.

The two-stage solution approach – where the first stage assigns machines to bays and the second stage determines a block layout for machines within each bay – has similarities to the approach employed herein to solve the DRLP-MMT. Another two stage approach is proposed by Tubaileh and Siam (2017) within the context of a multi-row layout problem. The first stage determines the sequence and relative position of the machines. The second stage finds the number of rows required to place all machines within the given space using simulated annealing and ant colony optimization.

Perhaps most related to our problem is the work presented by Solimanpur and Jafari (2008). In addition to machine replicas (with capacities) and multiple product flows, minimum clearance requirements in two dimensions are also considered. Unlike our work, however, the problem is not restricted to the double-row case. Instead, the authors consider the assignment of machines to the layout in two dimensions (i.e., (x, y) coordinates for the midpoint of each machine are determined). Another significant difference is that the referenced model restricts each machine to be dedicated to a single product. Such a restriction allows for a linear formulation, enabling a branch-and-bound-based solution technique.

Solimanpur and Kamran (2010) propose a genetic algorithm to solve a simplified version of the model described in Solimanpur and Jafari (2008). This simplified problem assumes that the layout is partitioned into equally-sized blocks (like the QAP). Taghavi and Murat (2011) propose a heuristic to solve the exact model formulated by Solimanpur and Jafari (2008). The proposed heuristic is based on decomposing the original problem into two simpler sub-problems and solving them separately in sequence. The first problem deals with machine-product assignment, where the flow sequences for each product across individual machines are chosen. The second sub-problem is the layout design problem where the location of each department (i.e., (x, y) coordinates) are determined. The sequential solution procedure continues until there is no further improvement. A similar approach is used by Wang et al. (2015) in solving dynamic double row layout problems. The flow is considered dynamic as it changes from one period to another, and the objective is to find the optimal layout over multiple periods.

There are also studies that consider network congestion as part of the layout design process. Ioannou (2007) restricts the maximum amount of flow on a link. Zhang et al. (2011) propose a flow assignment model with congestion to design layouts and flow routing decisions. A three-step function to represent varying capacity and travel time on each link is proposed. For a link between i and j there is another level l representing the capacity and the travel time. In the model, a continuous variable x_{ijl} represents the flow, while a binary variable y_{ijl} shows the choice of the capacity/travel time option. These works are more restrictive in terms of layout design. They do not allow subbranches that may split and then rejoin at any point of the network. The product goes through each machine type only once.

There is a category of DRLP problems in which the layout is modeled as quadratic assignment problem (QAP). QAP-based formulations assume that the facility may be partitioned into equally sized and shaped “cells”. For example, Urban et al. (2000) proposed the integrated machine allocation and layout problem (IMALP). Each product has a known demand and must visit a predefined sequence of machine types, such that each type has multiple machines. Each machine has a capacity, expressed in terms of available processing time. The layout is partitioned into a grid (i.e., a block layout), where it is assumed that each machine is identically-sized. There are no minimum-clearance restrictions. An optimal solution approach to the QAP formulation is presented, as well as three heuristic approaches (network/QAP, tabu search, and shortest path). Like our work, the IMALP allows each individual machine to process multiple products (other works restrict each machine to process a single product). Although multiple machine types are defined, each type may be visited at most once, thus prohibiting re-entrant flows. A similar approach was used by Amaral (2011), where mixed integer linear optimization is used for identically-sized departments.

Similarly, Jaramillo and McKendall (2010) present a generalized QAP formulation with multi-product flows. The machine pool includes different machine types with replicas, such that a subset of the machines must be allocated to a set of predefined locations. The problem includes two simultaneous decisions: assignment of machines to the floor and assignment of products to machines. A nonlinear mixed integer programming (MIP) formulation of the problem is developed and used for small problem instances. For large problems the authors propose a tabu search heuristic. The formulation does not allow alternative routes, and no products may visit the same machine type more than once.

More recently, Zhao and Wallace (2013) formulate the QAP with machine types and duplicate machines in each machine type category. A two stage solution approach is proposed. The allocation of machines to cells is determined in the first stage. In the second stage, the flow between individual machines is assigned. The approach is tested on deterministic and stochastic demand cases. For the stochastic case, five scenarios are used to describe the uncertain demand. Optimization is based on minimizing the total flow cost over five production scenarios.

Castillo and Peters (2003) consider a layout problem in which the department shapes are not assigned *a priori*. Instead, the layout area is divided into unit-sized identical grids, such that a combination of grids comprises an individual department. Space filling curves are used to create contiguous and connected departments. The formulation includes machine types and replicas. As in our work, machine capacities are defined at the replica level and the flow volumes between machine replicas are not determined *a priori*. A nonlinear MIP formulation is proposed, along with a decomposition-based heuristic approach. A two step process includes first solving the problem for flow allocation among machine replicas. In the second step, given the calculated flows from the first step, the number and shapes of the individual departments are determined. Although we are not aware of any application of decomposition approaches for the DRLP, such approaches have been successfully utilized in network flow problems, such as Campbell and Savelsbergh (2004), Romero and Monticelli (1994), Harjunkoski and Grossmann (2001), Yao et al. (2013).

The double-row layout optimization problem presented here requires simultaneous optimization of three sets of decision variables. The first set defines the relative order of machines in the upper and lower rows. As presented in Section 3.2, two binary decision variables, z_{rij} and y_{ir}, define the relative layout. Additionally, there are two sets of decision variables, one for machine locations and the other for flow quantities among machines. Even for a known layout (i.e., with fixed z_{rij} and y_{ir}), finding the optimal flow and machine locations is a challenging task. The cost includes the product of two decision variables, flow and distance, which leads to a nonconvex objective function. Such problems are called bilinear programming problems (BLP), with several well-known application areas in operations research. Network flow problems with splits and mixes or with nonconvex cost functions can be formulated as BLPs. For example, pooling problems from the petrochemical industry contain a combination of network flow and blending problems. In pooling problems, raw materials from input nodes are sent to output nodes (i.e., demand points) via a combination of intermediate mixing tanks. There are two sets of decision variables, flow and quality requirements of each demand point, and the objective is to meet to demand and quality requirements at

minimum cost. Gupte et al. (2016) describe the pooling problem with alternate formulations and discuss discretization methods to approximate the BLP as an MILP. Metaheuristics have also been used to solve pooling problems. Erbeyoglu and Bilge (2016) propose simulated annealing (SA) and particle swarm optimization (PSO) to solve larger instances of the problem. Another important application area for BLPs is Flexible Manufacturing System (FMS) loading problems. In a FMS, jobs can be processed on alternate machines with different sets of tools. The objective is to determine the tool selection and process assignment that minimizes material handling cost (i.e., flow cost). This requires simultaneous optimization of the flow and the tool/processes assignment. Kosucuoglu and Bilge (2012) propose a GA-LP heuristic to decompose the problem into two subproblems. The GA is used for tool allocation while the reduced problem is solved as an LP for routing decisions. In network flow problems, the cost function often is assumed to be linear. From a real-life application point of view this may not be an accurate representation of the cost. Nahapetyan and Pardalos (2007) studied a network flow problem with a piecewise cost function. They reformulated the problem as a BLP with linear constraints and a bilinear objective function.

One popular method to solve BLPs is an iterative procedure that utilizes a fundamental characteristic of such problems. As the objective function includes the product of two decision variables, fixing one of the variables makes the problem an ordinary LP. A starting solution is generated by selecting a feasible solution for one of the variables and then solving the LP for the second variable. The output is then fixed and the procedure is run to optimize the first variable. The iteration continues until convergence is achieved. Such solution approaches to quadratic problems are discussed in Audet et al. (2004). In our hierarchical solution approach, the machines are assigned to the upper and lower rows in the top level. Once the relative position of each machine is determined, the problem is formulated and solved as BLP in the lower level by following the methodology described in Audet et al. (2004).

3. Problem definition

The DRLP-MMT may be formally defined as follows. Let T represent the set of all machine types, such that each type describes a classification of processing capabilities. We define M_t to denote the set of all replicas of a particular machine type $t \in T$. Each individual machine is given a unique identification number, such that $I = \cup_{t \in T} M_t$ provides the set of all machines. We specify the width of machine $i \in I$ as w_i, and the processing capacity (the number of products that may be produced per unit time) of machine i as u_i. Machine capacities are fixed, and may reflect expected downtime, but we do not explicitly consider queueing or scheduling.

Let P represent the set of all products to be produced in the facility. Each product $p \in P$ has a known demand of d_p items per unit time, and must visit a predefined sequence of machine types (classes) in the production process. The process flow for product $p \in P$ is given by the set S_p, such that the last element of each set S_p is a "dummy" machine type. This is a virtual sink node in the network, which does not correspond to an actual physical machine, and denotes the end of a process. It is permissible for the same machine type to appear multiple times for a given S_p.

The layout is confined to two rows of machines (an upper and a lower row), given by the set $R = \{U, L\}$. The distance separating the two rows is given by the scalar parameter c. Individual machines shall be placed along the two rows such that adjacent machines $i \in I$ and $j \in \{I \backslash i\}$ must be separated by a clearance of at least a_{ij} distance units. A summary of the parameter notation is provided in Table 1.

The objective of the DRLP-MMT is to minimize the overall material handling cost, which is defined to be the distance traveled between machines times the quantity of products that are moved between those machines. We assume that the material handling system operates via rectilinear travel between machines, such that machines in the *same* row do not incur a vertical travel distance. The load/unload ports on each machine are assumed to be located at the center point of the machine (i.e., at a point located at one-half of the machine's width). For each product the model requires only the demand (in number of units) and the machining sequence as input. How each product flows in the layout and how production splits among replicate machines are determined by the optimization process.

In determining the minimum-cost layout, the following decisions are required. First, the continuous decision variable $x_{ir} \geq 0$ represents the absolute location (position) of each machine replica $i \in I$ in row $r \in R$; $x_{ir} = 0$ if machine i is not placed in row r. Next, while the sequence of machine *types* is given, the optimal flow among individual machine *replicas* must be determined. The integer decision variable f_{psij} represents the quantity of product $p \in P$ that should flow from machine replica i to replica j when i is of the sth machine type in product p's process flow. Specifically, f_{psij} is defined for all $p \in P$, $s \in \{1, \ldots, |S_p| - 1\}$, $i \in M_{S_p(s)}$, and $j \in M_{S_p(s+1)}$, where $S_p(s)$ represents the machine type associated with the sth step in the process flow of product p. The example below illustrates the terminology. Other supplementary decision variables required for the mathematical programming formulation of this problem are summarized in Table 2.

3.1. Example

A small example is provided to show the notation defined above. Here, we consider a manufacturer of two products ($P = \{1, 2\}$) with known demands of $d_1 = 30$ and $d_2 = 50$ units per hour. The designation of the time interval is unimportant, provided that it is consistent with the time units employed by the machine capacity parameters. For example, if the production rate is expressed in terms of units per hour, the capacity should also be expressed as units per hour.

There are three machine types, given by $T = \{A, B, C\}$, such that machine type A has two replicas, machine type B has three replicas, and machine type C has two replicas. Given that each replica

Table 1
Parameter notation.

T	Set of machine types.
M_t	Set of individual machine replicas of type $t \in T$.
I	Set of all distinct individual machine replicas, such that $I = \cup_{t \in T} M_t$.
u_i	Capacity (upper bound of production) for machine replica $i \in I$.
w_i	Width of machine replica $i \in I$.
a_{ij}	Minimum clearance required between machine replicas $i \in I$ and $j \in \{I \backslash i\}$.
P	Set of products to be produced.
d_p	Demand for product $p \in P$.
S_p	Sequence of machine types that must be visited to produce product $p \in P$.
R	Set of rows in the layout, such that $R = \{U, L\}$ specifies the upper and lower rows.
c	Width of the aisle (corridor) separating the two rows.

Table 2
Decision variable definitions.

x_{ir}	Continuous variable representing the mid-point location of machine replica $i \in I$ in row $r \in R$. $x_{ir} = 0$ if machine i is not placed in row r.
y_{ir}	Binary variable, defined for all $i \in I$ and $r \in R$, where $y_{ir} = 1$ if machine i is assigned to row r ($y_{ir} = 0$ otherwise).
z_{rij}	Binary variable, defined for all $r \in R$, $i \in I$, $j \in \{I \setminus i\}$, such that $z_{rij} = 1$ if machines i and j are placed in row r and j is located somewhere to the right of i.
f_{psij}	Flow from machine replica $i \in I$ to $j \in I$ at sequence $s \in \{1, \ldots, \lvert S_p \rvert - 1\}$ for product $p \in P$.
q_{ij}	Auxiliary binary variable; $q_{ij} = 1$ if replicas $i \in I$ and $j \in \{I \setminus i\}$ are assigned to the same row, $q_{ij} = 0$ otherwise.
v^+_{ij}, v^-_{ij}	Auxiliary continuous variables used in linearizing the distance metric, $\lvert x_{ir} - x_{jr} \rvert$

Table 3
Machine replica production capacities and widths for the small-scale example.

Type	A		B			C	
Replica, i	1	2	3	4	5	6	7
Capacity, u_i	70	80	55	50	40	65	65
Width, w_i	2.5	2.0	3.0	2.5	2.5	3.5	3.0

Table 4
Values of the f_{psij} decision variables for the example solution.

Product 1		Product 2	
$f_{1,1,3,6} = 30$	$(B \to C)$	$f_{2,1,6,1} = 35$	$(C \to A)$
$f_{1,2,6,1} = 30$	$(C \to A)$	$f_{2,1,7,2} = 15$	
$f_{1,3,1,7} = 30$	$(A \to C)$	$f_{2,2,1,4} = 35$	$(A \to B)$
$f_{1,4,7,5} = 30$	$(C \to B)$	$f_{2,2,2,4} = 15$	
$f_{1,5,5,8} = 30$	$(B \to$ sink$)$	$f_{2,3,4,2} = 50$	$(B \to A)$
		$f_{2,4,2,8} = 50$	$(A \to$ sink$)$

must be assigned a unique identification number, we define $M_A = \{1, 2\}$, $M_B = \{3, 4, 5\}$ and $M_C = \{6, 7\}$. Thus, our set of all machine replicas is given by $I = \{1, 2, \ldots, 7\}$. Table 3 contains the production capacity (units per hour) and width (meters) of each replica.

Product 1 must visit the machine types in the following order: $B \to C \to A \to C \to B$. Product 2's process flow is given by: $C \to A \to B \to A$. Thus, after adding the dummy sink node, we have

$S_1 = \{B, C, A, C, B, \text{sink}\}$, and

$S_2 = \{C, A, B, A, \text{sink}\}$.

Note that, given the product demands and the replica production capacities, it is necessary for some machines to process both products. Furthermore, it is necessary for at least one of the products to follow multiple paths through the facility, that is, the flow must be split. One feasible, but not necessarily optimal, process flow is provided in Fig. 1. The corresponding values of f_{psij} are summarized in Table 4. In Fig. 1, the flow route for product 1 is shown as a solid line. The first product starts at machine 3 and then visits machines 6, 1, 7 and 5. All 30 units of this product are processed on a single line. Product 2 starts on two separate branches. The first branch produces 35 units and includes machines 6, 1 and 4 (shown as long dashed lines in Fig. 1). The second branch (a dotted line) has a production volume of 15 units and uses machines 7, 2, 4. Two separate branches of the product 2 join together at machine 4. Then, product 2 proceeds to the next machine (i.e., machine 2) as a single line.

3.2. Mathematical programming formulation

A mixed integer nonlinear programming formulation of the MRLP-MTT may be established as follows.

$$\text{Min} \quad \sum_{p \in P} \sum_{s=1}^{|S_p|-2} \sum_{i \in M_{S_p(s)}} \sum_{j \in M_{S_p(s+1)}} f_{psij}\left(v^+_{ij} + v^-_{ij} + c(1 - q_{ij})\right) \tag{1}$$

$$\text{s.t.} \quad x_{ir} \le M y_{ir}, \quad \forall\, i \in I, r \in R, \tag{2}$$

$$\sum_{r \in R} y_{ir} = 1 \quad \forall\, i \in I, \tag{3}$$

$$\frac{w_i y_{ir} + w_j y_{jr}}{2} + a_{ij} z_{rji} \le x_{ir} - x_{jr} + M(1 - z_{rji}) \quad \forall\, i \in I, j \in \{I \setminus i\}, r \in R, \tag{4}$$

$$\sum_{r \in R} x_{ir} - \sum_{r \in R} x_{jr} = v^-_{ij} - v^+_{ij} \quad \forall\, i \in I, j \in \{I \setminus i\}, \tag{5}$$

$$z_{rij} + z_{rji} \le \frac{y_{ir} + y_{jr}}{2} \quad \forall\, i \in I, j \in \{I \setminus i\}, r \in R, \tag{6}$$

$$z_{rij} + z_{rji} + 1 \ge y_{ir} + y_{jr} \quad \forall\, i \in I, j \in \{I \setminus i\}, r \in R, \tag{7}$$

$$q_{ij} = \sum_{r \in R} (z_{rij} + z_{rji}) \quad \forall\, i \in I, j \in \{I \setminus i\}, \tag{8}$$

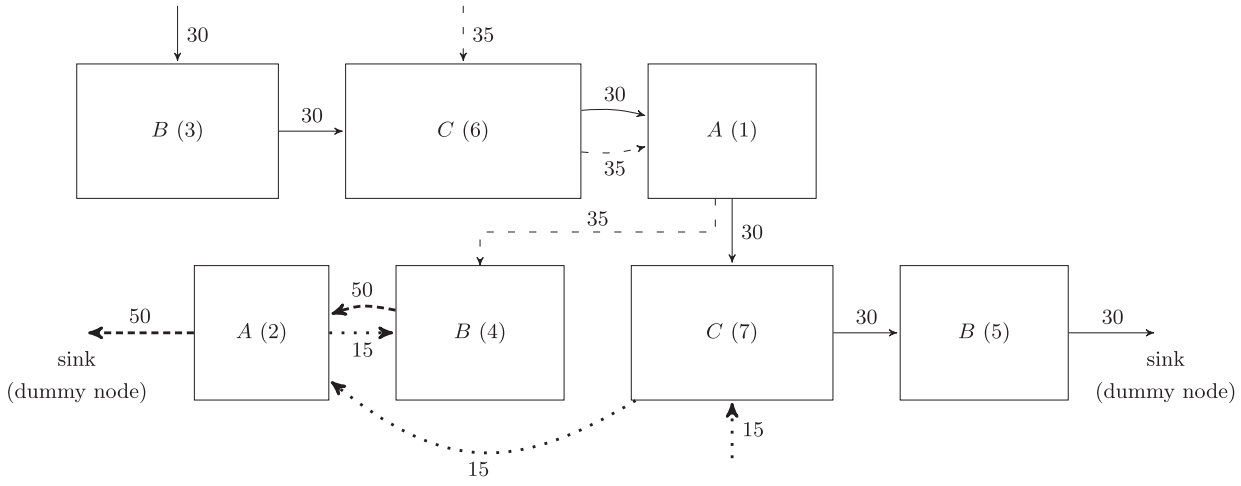


Fig. 1. Depiction of material flow quantities in a feasible layout for the example problem.

$$\sum_{i \in M_{S_p(s)}} \sum_{j \in M_{S_p(s+1)}} f_{psij} = d_p \quad \forall\ p \in P, s \in \{1, \ldots, |S_p| - 1\}, \tag{9}$$

$$\sum_{i \in M_{S_p(s-1)}} f_{p,s-1,ij} = \sum_{k \in M_{S_p(s)}} f_{psjk} \quad \forall\ p \in P, s \in \{2, \ldots, |S_p| - 1\}, j \in M_{S_p(s)}, \tag{10}$$

$$\sum_{p \in P} \sum_{i \in M_{S_p(s)}}^{s=1} \sum_{j \in M_{S_p(s+1)}}^{|S_p|-1} f_{psij} \le u_i \quad \forall\ i \in I, \tag{11}$$

$$f_{psij} \in \{0, 1, \ldots\} \quad \forall\ p \in P, s \in \{1, \ldots, |S_p| - 1\}, i \in M_{S_p(s)}, j \in M_{S_p(s+1)}, \tag{12}$$

$$v_{ij}^{+}, v_{ij}^{-} \ge 0 \quad \forall\ i \in I, j \in \{I \setminus i\}, \tag{13}$$

$$x_{ir} \ge 0 \quad \forall\ i \in I, r \in R, \tag{14}$$

$$y_{ir} \in \{0, 1\} \quad \forall\ i \in I, r \in R, \tag{15}$$

$$z_{rij} \in \{0, 1\} \quad \forall i \in I, j \in \{I \setminus i\}, r \in R. \tag{16}$$

The nonlinear objective function (1) seeks to minimize the total cost of material handling. The upper limit in the second summation, $|S_p| - 2$, serves to exclude flow to the dummy sink node. The terms within parentheses determine the rectilinear distance between machines i and j. Constraint (2) defines $x_{i,r} = 0$ if machine i is not placed in row r, while Constraint (3) ensures that each machine is placed in exactly one row. Clearance requirements between machines are addressed in Constraint (4). The M in (2) and (4) represents a big number. Since both constraints are related to the locations of machines, a number that is larger than the width of the longest possible layout could be used as the M value. The sum of widths for all machines plus the sum of minimum clearances is larger than the width of the longest layout, and is a reasonable approximation for M.

Constraint (5) establishes proper values for $v_{i,j}^{+}$ and $v_{i,j}^{-}$, which determine the horizontal distance between machines i and j. Constraints (6) and (7) establish the relationships among 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} must equal one; otherwise, $z_{rij} = z_{rji} = 0$. Constraint (8) determines whether machines i and j are placed in the same row.

Product flow requirements are addressed in Constraints (9)–(11). In particular, Constraint (9) ensures that the demand for product p is met by processing the appropriate quantity in each step of the process. Conservation of flow is maintained in Constraint (10), which states that the quantity of product arriving at machine j must equal the quantity departing from that machine. Production capacity limitations are enforced by Constraint (11). Finally, Constraints (12)–(16) provide the decision variable definitions.

4. Solution approach

The mathematical formulation of the DRLP-MMT, presented in Section 3.2, has a nonlinear objective function that includes products of decision variables. Since there is no exact method to solve the problem optimally, a two-step heuristic approach is proposed, whereby the problem is decomposed into two hierarchical components. Similar approaches were previously used in solving various facility layout problems such as in Montreuil et al. (2004) and Kulturel-Konak (2012). In the first phase of our approach, the number of machines placed in the upper and lower rows, as well as the relative sequencing of these machines within each row, is determined. Thus, Phase I establishes the values of decision variables y_{ir}, z_{rij}, and q_{ij}. The second phase determines the *absolute* positioning of machines and the quantity of flow among machine replicates. In other words, Phase II establishes the values of the remaining decision variables, x_{ir} and f_{psij}.

Fig. 2 shows a flowchart for the solution approach. Phase I begins with a randomly generated single layout, L_0, which is defined as a sequence of machines in the upper and lower rows. Using this initial layout as a starting point, n alternative layouts are generated (the methodology used to generate these layouts is specified in Section 4.1). Once the layouts are created, the algorithm of Phase II is employed to determine the exact machine positions and the flow quantities that minimize the total transportation cost. The layouts are then sorted by descending cost and the best layout (i.e., minimum total cost) is kept for the next iteration. The best layout of the current iteration becomes L_0 of the next iteration. The iterative procedure continues until convergence. Algorithm 1 shows the pseudo-code of the solution approach.

input : L_0 Initial layout: sequence of machines randomly assigned to upper and lower rows
output: L^* Best layout with minimum transportation cost, f^*_{psij} optimum flow, x^*_{ir} optimum locations of machines
while *convergence criteria is not satisfied* **do**
  PHASE I: Generate n alternative layouts from L_0
  for $i \leftarrow 1$ **to** n **do**
    select a random number of machines n_r to remove from L_0;
    remove n_r machines from L_0 in random order. The remaining layout is L_r (reduced layout);
    insert n_r machines to L_r to create L_i;
    For L_i run Phase II;
    PHASE II: Calculate optimum flow (f_{psij}) and machine locations (x_{ir});
    initialization: set x_{ir} values based on minimum clearance requirements;
      $indx = 0$;
    while $((F^{II}_{indx} < F^{II}_{indx-1})$ *or* $(indx \le 1))$ **do**
      Sub Problem I: Given x_{ir}, optimize for f_{psij}; Minimized flow cost: F^{I}_{indx};
      Sub Problem II: Given f_{psij}, optimize for x_{ir} ; Minimized flow cost: F^{II}_{indx};
      $indx + 1$;
    end
  end
  order n layouts (L_i, $i = 1$ to n) by flow cost;
  select the layout with minimum flow cost (L^*);
  $L_0 \leftarrow L^*$;
end

Algorithm 1: Pseudo-code of the solution approach.

In our test runs the number of alternative layouts created at each iteration (n) is set to 20. This provides a sufficiently large search neighborhood with a diverse set of alternative layouts. In our test runs, the iterative procedure for a better layout is terminated if there is no improvement (i.e., reduction in total transportation cost) within 200 iterations.

4.1. Phase I

The primary function of Phase I is to create a *search neighborhood* in the form of alternative layouts. These layouts are generated by randomly removing and then inserting machines from a single

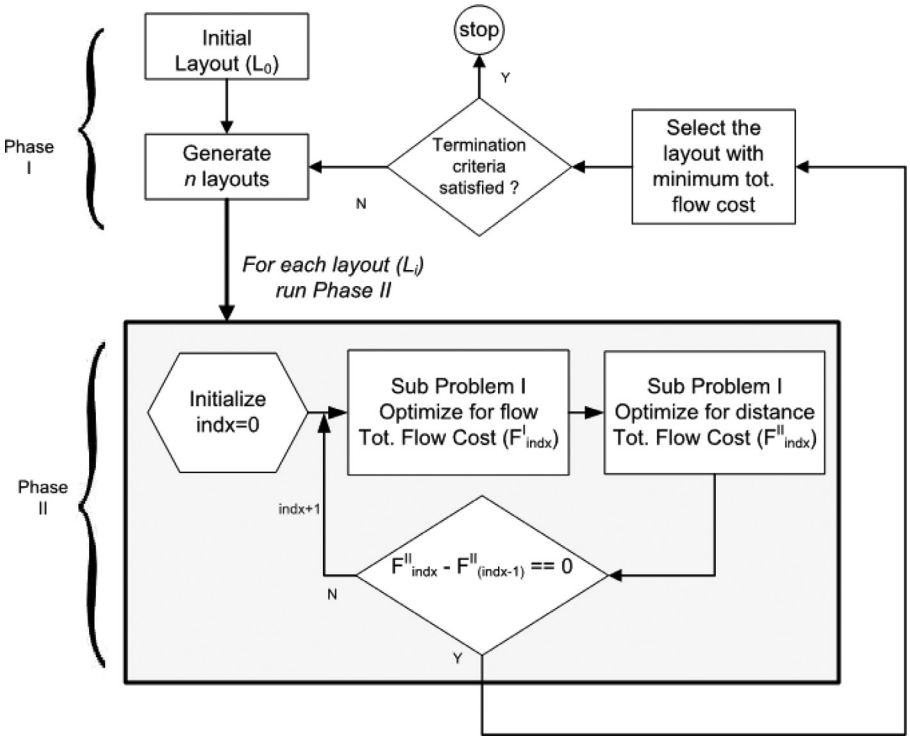


Fig. 2. Flowchart for the solution approach.

starting layout. The steps of the procedure are presented below. Fig. 3a–e illustrate the process with a small example.

Phase I starts with an initial layout, L_0, which is constructed by assigning the available machines to the upper and lower rows randomly. Fig. 3a shows a starting layout with eight machines; machines 1 through 5 in the upper row, and machines 6, 7, and 8 in the lower row.

To create new layouts from L_0, some machines are removed from L_0. The number of machines that are to be removed is determined by n_r, which is drawn from a uniform discrete distribution between 0% and 80% of the total number of machines. This means that up to 80% of the machines can be removed from a given layout. A sufficiently high percentage allows the creation of alternatives that are much different from the current layout. If the upper limit is set to a lower value (i.e., less than 50%), alternatives have less variability, and slow or premature convergence to a local optimum.

The machines for removal are selected randomly from the initial layout (see Fig. 3b). Fig. 3c shows the "reduced layout" after six machines are removed from the initial layout. Note that there are no machines left in the lower row after these particular removals. The upper row includes the remaining two machines (2 and 3). The list of removed machines is given on the right-hand side of Fig. 3c. Machine 8 is the first one removed from the layout, followed by machines 4, 5, 1, 7, and 6.

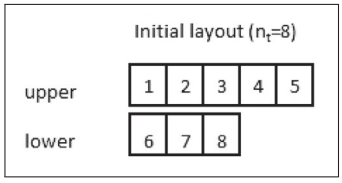
A new layout is created by inserting previously removed machines into the "reduced layout". Fig. 3c shows a reduced layout with machines 2 and 3 in the upper row and an empty lower row. The insertion process involves recursively adding previously removed machines back to the base layout. Starting with the reduced layout as an initial base, the process creates a new base after each insertion. The insertion point of a machine to a layout is selected randomly. This process continues until all removed machines are inserted back into the layout.

To simplify the insertion process a temporary array is used to store the upper and lower machines, along with starting (S), ending (E), and upper-lower row break point (BR) markers. Fig. 3d shows four potential insertion points (I1 to I4) on the temporary array. Keeping all relevant information of the layout in a one-dimensional array simplifies the coding of the insertion process. In the example, machine 8 is the first to be inserted into the reduced layout. Fig. 3e shows all four possible intermediate layouts that could be created based on the insertion point, and one of these intermediate layouts is selected randomly. After each insertion, the intermediate layout is used as a new base to insert the next machine in line (i.e., machine 4 in the example). This process continues until all removed machines are inserted into the reduced layout. A wide variety of layouts can be generated from an initial single layout using this approach.

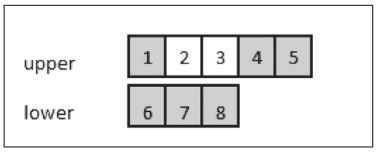
4.2. Phase II

The process described in Phase I is used to generate new layouts which are defined as sequences of machines along the upper and lower sides of the row. According to the DRLP-MMT formulation given in constraints (1)–(16), three decision variables, y_{ir}, z_{irj}, and q_{ij} are selected in Phase I. The reduced problem is sent to Phase II to calculate the values of the other two decision variables, x_{ir} and f_{psij}, in order to minimize total flow cost. Flow cost is expressed as a product of flow and distance between two machines. A product of two decision variables in the objective function does not allow a transformation into a linear form, making the problem unsolvable as a MILP. A solution approach defined in Audet et al. (2004) uses an iterative heuristic that divides decision variables into two subsets. Fixing either subset of the decision variables reduces the problem to a simple LP. The iterative procedure involves solving the two simple LPs alternately until convergence.

We use a similar approach that solves the problem in two steps. Our implementation of this procedure begins with the assumption that the gap between each adjacent pair of machines is equal to the minimum required clearance. We also assume that there is no space between the left-most machine and the wall. These are

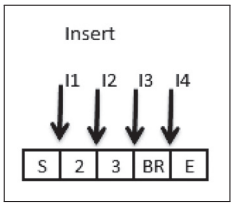


(a) Initial layout, L_0

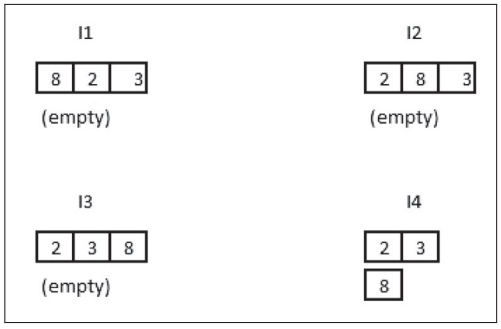


(b) Six machines are selected for removal (n_r = 6)

(c) Reduced layout after six machines are removed from L_0



(d) Potential insertion points for machine 8



(e) Four potential intermediate-layouts after inserting machine 8 to the reduced layout

Fig. 3. Example of layout generation process.

feasible starting values for the distance related variables, x_{ir}. With the distance variables fixed, the problem can be solved for the optimal flow. Next, these flow values are kept constant and the problem is solved for the optimal distance. When there is no improvement in either problem the iterative procedure is terminated.

5. Computational experience

The double row layout problem in our study includes fewer restrictions in terms of layout structure than previous work in this area. There are no test problems in the literature that could be used for comparison, as this is a more realistic and difficult problem. Thus, to assess the performance of the proposed approach, test problem instances were generated according to three parameters: number of machine types, number of replicates for each machine type, and number of products.

5.1. Test problems

Three levels of number of machine types were considered (5, 10, and 20 machine types). For the case of five machine types, the number of replicates for each type was randomly chosen to be between 1 and 4; For 10 (20) machine types, the number of replicates was between 1 and 3 (1 or 2). The width of each machine replicate, w_i, was uniformly selected between 1.0 and 2.5 distance units, with required clearances between replicates, a_{ij}, uniformly distributed between 0.25 and 0.75. The size of the corridor separating the upper and lower rows, c, was uniformly distributed between 0.25 and 1.25 units. Four levels of number of product types were evaluated (1, 2, 5, or 10), with demand for each product type uniformly chosen between 100 and 500 units. For each product type, the number of machine types comprising a process flow was uniformly distributed between 40–60%, 90–110%, or 140–160% of the number of machine types. The number of machines each product visits during the production sequence represents the process flow length. Three replications for each combination of parameter setting lead to a total of 108 test problem instances. Table 5 summarizes the parameter combinations. (All test problem data will be made available in an on-line supplement to the published paper). Because of hardware updates and software availability, we have used different platforms. For the base cases given in Table 6, we used a Linux 64 server with Intel Core2 Quad CPU @2.66 GHz. The coding is in C++ and the CPLEX 12.6 solver engine is used for optimization. The COIN-OR MINLP runs are performed on a windows-based machine with i5-7200 CPU @2.50 GHz (4 CPUs) and 8 GB RAM hardware.

5.2. Performance

In evaluating the outcomes of the test problems we focus on two aspects: solution quality and convergence speed (how fast the problem can be solved). Due to the nonlinearity of the problem, we have limited options for a one-to-one comparison with any other standard methodology. As an alternative assessment we ran with different initial conditions (i.e., different starting layouts) and then checked the consistency of the final results. Although this does not provide provable information about the solution quality in terms of closeness to the optimal, a consistent convergence

Table 5
Parameter settings used to generate test problem instances.

				units
Machine types	5	10	20	number
Machine replicates	$\sim$U(1, 4)	$\sim$U(1, 3)	$\sim$U(1, 2)	number
Machine width	$w_i \sim$U(1.0, 2.5)			distance units
Clearance	$a_{ij} \sim$U(0.25, 0.75)			distance units
Corridor width	$c \sim$U(0.25, 1.25)			distance units
Product types	$\|P\| \in \{1, 2, 5, 10\}$			number
Demand	$d_p \sim$U(100, 500)			number of units
Process flow length	$\|S_p\| \sim$U(0.4, 0.6) $\times \|I\|$, $\|S_p\| \sim$U(0.9, 1.1) $\times \|I\|$, $\|S_p\| \sim$U(1.4, 1.6) $\times \|I\|$			number of machines

Table 6
Summary statistics for the test cases .

Group	Machine types	Number of products	Average production sequence length	Average number of machines*	Average time to convergence [min]*
1	5	1	5.333	14.000	5.259
2	5	1	6.667	14.000	8.656
3	5	1	8.000	13.000	9.042
4	5	2	3.000	12.667	7.701
5	5	2	5.167	12.000	7.127
6	5	2	7.333	13.333	12.288
7	5	5	2.467	10.667	7.256
8	5	5	5.000	13.667	23.231
9	5	5	7.400	12.333	20.882
10	5	10	2.500	11.333	12.710
11	5	10	5.000	12.333	26.167
12	5	10	7.600	13.000	37.914
13	10	1	10.667	17.667	15.625
14	10	1	13.000	20.333	23.060
15	10	1	17.000	17.667	16.103
16	10	2	6.833	21.000	26.813
17	10	2	10.333	17.667	18.089
18	10	2	14.833	20.000	46.151
19	10	5	5.267	19.667	35.466
20	10	5	9.933	18.000	44.578
21	10	5	15.000	22.333	135.993
22	10	10	5.000	20.333	117.675
23	10	10	9.833	19.667	137.637
24	10	10	14.967	20.333	302.476
25	20	1	20.667	30.000	43.385
26	20	1	26.333	26.333	71.419
27	20	1	33.333	30.667	156.034
28	20	2	12.667	30.000	124.675
29	20	2	21.000	30.667	212.022
30	20	2	30.333	31.667	233.973
31	20	5	10.333	29.333	263.723
32	20	5	20.267	29.333	386.667
33	20	5	30.067	30.333	884.354
34	20	10	10.367	30.667	630.967
35	20	10	19.867	29.333	1282.548
36	20	10	30.200	29.667	1327.329

(*) Based on three replications.

pattern is an indicator of the robustness of the solution approach. In addition to solution quality, another important criterion is the convergence speed. Usually there is a positive correlation between the size of the problem and the solution time. It is important to understand how different parameters impact the solution time. The number of machine types, the number of products, and the average process flow length per product (i.e., average number of machines visited) are the most important parameters determining the size of DRLP problems. Analysis of these parameters and convergence speed is performed for the test cases. Table 6 includes summary statistics for the test cases. Two performance measures, total number of iterations to convergence and run time, are presented in the table. The data are grouped by number of machine

Table 7
ANOVA analysis of test cases.

Response: run time (y)					
	df	Sum Sq	Mean Sq	F value	Pr($>F$)
# machine types	1	4,165,202	4,165,202	104.688	2.200e-16
# products	1	2,527,075	2,527,075	63.515	2.145e-12
interaction	1	2,890,751	2,890,751	72.656	1.302e-13
residuals	104	4,137,837	39,787		

and product types and process flow length with the following levels:

- Three levels of machine types: 5,10, and 20
- Four levels of product types: 1, 2, 5, and 10
- Three levels of process flow lengths: three different uniform distributions (see Table 5)

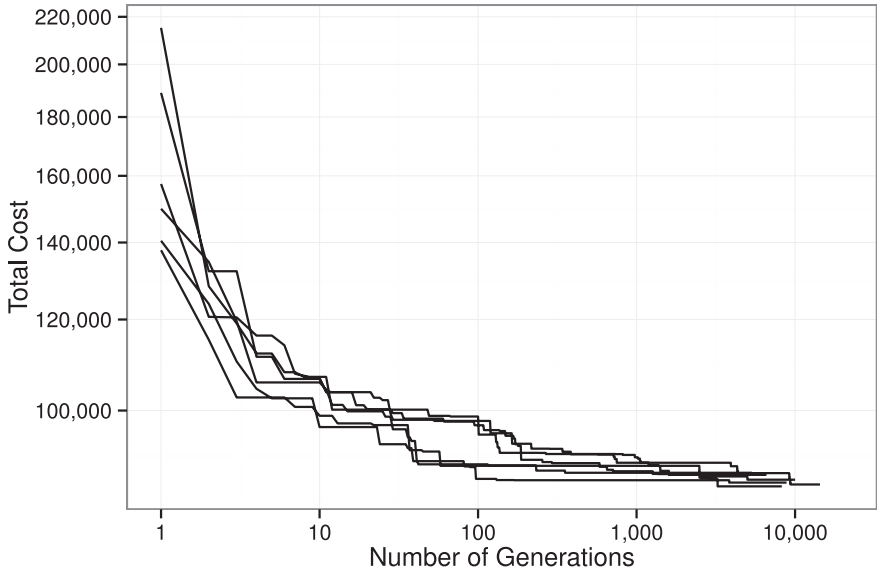
There are $3 \times 4 \times 3 = 36$ groups as per above. For each group, three instances of process flow length are generated from the uniform distributions given in Table 5. The number of iterations and runtime to convergence are based on the average value of the three instances. For most cases convergence is achieved in minutes or in several hours. In a few extreme cases where 10 different products must visit up to 30 machines, the convergence extends up to 33 h. As expected, convergence time increases as the number of machine types and number of products increase. Since we solve the layout problem for design purposes, the computational speed has less importance as long as it is tractable. The design problem will be solved infrequently.

Table 7 shows a two-way ANOVA analysis of the 108 test cases. Two main effects – the number of machine types and the number of products – are significant, with p-values less than 0.01. The interaction between the two main effects is also significant.

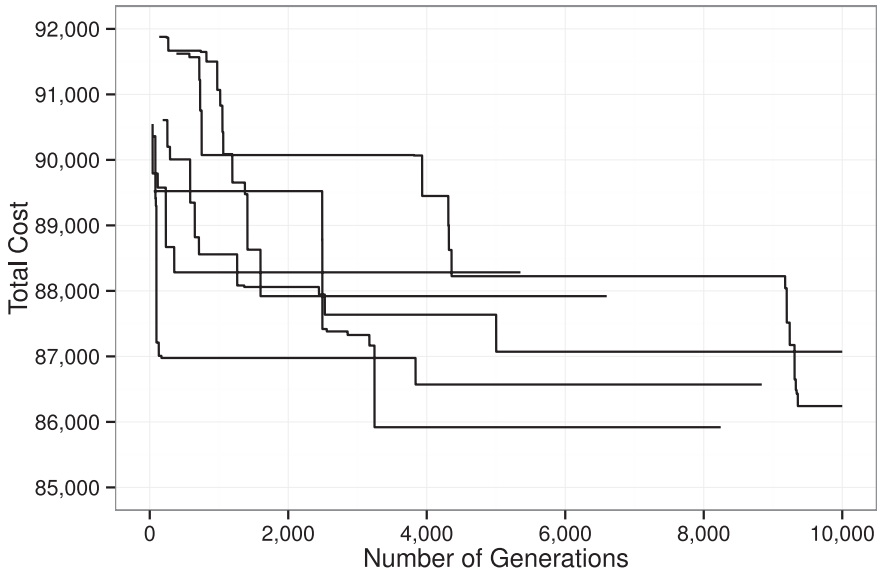
5.3. Robustness of the proposed approach

In terms of quality of the final solutions and the robustness of the proposed approach, we examined the consistency of the results under different starting conditions. For this purpose we selected a test case, and ran the algorithm six times with a changing initial random seed. The selected test case is a large problem with 10 machine types and 10 products. The average process length for 10 products is 9.7 machines. As explained in Section 3, each product is required to visit the set of machines in a predetermined sequence. The list of machines are independently generated for each product. The process length is defined as the number of machines each product has to visit.

Fig. 4a and b show the convergence pattern for the test case. In most runs we reach the convergence region within a few thousand iterations, as can be seen in the *log x - log y* graph of Fig. 4a. To see the spread of the final values we zoom into the convergence region, which is given in Fig. 4b. The final values are between 86,000



(a) Full convergence pattern of six runs



(b) Zoom in view of the convergence region

Fig. 4. Convergence pattern for test case #2.

and 89,200, and these correspond to a spread of 3.7%, in terms of percent deviation of the worst solution from the best one. Given the complexity of the problem, this very small diversity between the worst and best solutions indicates a very stable, effective optimization procedure.

Table 8 shows convergence variation for 14 additional test cases. To get a better insight into convergence, the number of replications for each case is increased to 10. Based on 10 runs, the mean and the standard deviation are calculated for each case. The ratio of standard deviation to median, similar to coefficient of variation, is an indicator of spread among runs. Another version of this metric is calculated by using only the five best solutions (i.e., lowest cost) of each case. Keeping only better solutions is expected to reduce the impact of initial random solutions on convergence quality. Although spreads get wider with an increasing number of products, machine types and machines, the ratio remains below 3%.

5.4. Computational performance with respect to benchmarks

To test the performance of the proposed method, we compare the outcome of the proposed method to three benchmark approaches. The first benchmark consists of one million "intelligently" randomly generated solutions. In the second benchmark, we replace the first part of our hierarchical algorithm with a tabu search algorithm. The last benchmark is a nonlinear mixed integer programming (MINLP) solver. We used COIN-OR Couenne to solve some of the smaller cases of Table 6. Finally, we consider the impact of flow splitting on solution quality. The flow splitting can allow improved utilization by balancing workload across machines.

Table 8
Convergence statistics for test cases .

Test case	Machine types	# of machines	# of products	$\left[\frac{\text{std}}{\text{median}}\right]^{\star}$	$\left[\frac{\text{std}}{\text{median}}\right]^{\dagger}$
1	5	11	5	0.03%	0%
2	5	12	2	0.46%	0%
3	5	12	10	1.43%	0.02%
4	5	14	10	0.53%	0.27%
5	5	14	2	0.95%	0.97%
6	5	15	1	1.64%	0.32%
7	5	15	1	1.43%	0.46%
8	5	15	5	1.41%	1.06%
9	10	15	1	2.55%	0.37%
10	10	19	10	2.68%	1.40%
11	10	20	2	3.44%	1.70%
12	10	22	2	4.32%	0.68%
13	10	22	5	4.06%	0.64%
14	10	22	5	3.10%	2.36%

(⋆) Based on 10 runs - (†) Based on the lowest 5 runs.

Furthermore, it expands the solution space, which could potentially afford the existence of lower-cost solutions.

5.4.1. Intelligent randomly generated solutions benchmark

As explained in Section 3 (Problem Definition), the solution approach is based on simultaneous optimization of three sets of decision variables:

- Layout – y_{ir}, z_{rij}, q_{ij};
- Distance between machines – x_{ir};
- Flow between machines – f_{psij}.

If the first two sets of decision variables are fixed, the mathematical model given in (1)–(16) is reduced to a simple LP and can be solved for optimal flow (i.e., the third set of variables). By using this principle we follow a three-step process to intelligently generate random solutions. For the test cases given in Table 9 we first generate one million random layouts. These layouts specify the order of machines along the upper and lower sides of the row. As a second step the specific location of each machine on the x-axis is selected randomly. Initially the gap between two adjacent machines is set to the minimum required distance. Then, random perturbations ranging from 0 to 100 percent of the minimum distance requirement are added to the minimal gaps. Once the order and the x-axis location of each machine is fixed, the problem is solved as an LP for the optimal flow between machines in the final step. Table 9 gives the minimum of one million such solutions for each case. For the random solution, the table shows the best among one million intelligently generated random solutions. As can be seen from the table, the intelligent random solution is the worst.

5.4.2. Tabu search benchmark

Tabu Search (TS) is a general heuristic procedure that use adaptive memory to create flexible search behavior. It uses tabu lists to avoid cycling back to previously-visited solutions based on a selected strategy. TS has been successfully applied to many combinatorial problems, and it is one of the popular techniques used in finding good solutions to large combinatorial problems in practical settings. In the area of facility planning, TS is used by McKendall and Jaramillo (2006), McKendall (2008), Samarghandi and Eshghi (2010), and da Silva et al. (2012), for example. In our implementation, the TS starts with a randomly-generated single layout. A neighboring set of layouts is created from the starting layout by insertion. The initial layout information is saved in a one-dimensional array with the ID's of the upper and lower row machines with a marker in the middle to identify the break point between the rows. First, a machine from the layout is randomly selected and then removed from the layout. If there are$n$ machines in the layout, each machine from 1 to n has an equal chance of removal. Once the machine is removed from the layout, the gap is filled by pushing machines on the right-hand side of the removal point to one space left. This creates a new, but temporary, layout with *n-1* machines. In the second step, the removed machine is inserted into this temporary layout. The insertion point is selected randomly, and it could be to the left or to the right of any machine in the layout. The breaking marker between the rows is treated as a machine so that insertion could be made to the beginning or end of either row.

The randomly created layouts constitute the search neighborhood. The neighborhood size is set to 20, which is equal to the neighborhood size used in the test runs of Table 6. For comparison purposes, we use identical parameters to maintain the same level of search complexity between the original approach and the TS. The tabu list includes the machine types that are temporarily excluded from the removal/insertion process. The size of the tabu list depends on the number of machine types in a case. For the cases with five machine types, the size of the tabu list is set to two. For ten machine types, we keep a tabu list of five machines. The list is updated on a FIFO basis every 20 iterations.

The termination criterion is based on two parameters. If there is no improvement after a priori-set number of iterations, the search is terminated. There is also a limit for the total number of iterations. In our test runs, the search is terminated if there is no improvement for 200 iterations, or if the total number of iterations reaches 20,000. Before the runs, we tested several TS parameters including different list sizes and different move operators in an ad-hoc manner. We noticed that the TS exhibits robust performance under changing parameters.

Table 9 shows average total flow costs for the proposed and the two benchmarks. As shown in this table, the proposed method produces better results in all five test cases. The average flow cost for the proposed method and the tabu search are based on 10 replications. A two-sample t-test is performed to determine if there is a significant difference between the average flow cost of the proposed method and TS. The p-values listed in the last column of Table 9 are sufficiently low to indicate a significant difference in average flow costs at $\alpha = 0.05$ for all test cases.

Table 9
Performance comparison: proposed approach versus tabu search and the best of one million intelligent random solutions.

Case No.	Number of machine types	Number of total machines	Number of products	Average process length	**Total flow cost**			Two sample t-test Proposed vs. Tabu p-value
					Proposed method	Tabu search	Random solution	
1	5	12	1	5.0	3,547.6	3,739.9	3.980.7	0.0057
2	5	12	5	2.6	3,686.5	3,715.4	3,943.5	0.0286
3	5	12	10	2.6	10,353.9	10,436.4	10,881.6	0.0134
4	10	16	1	11.0	5,577.2	5,652.7	7,318.2	0.0005
5	10	22	5	5.2	17,898.9	18,265.7	22,953.3	0.0002

Table 10
MINLP runs.

Test case	Number of machine types	Number of total machines	Number of products	MINLP			Proposed Approach	
				Total Flow C. without initial soln	Total Flow C. with initial soln	Run time (min)	Flow cost	Run time (min)
6	5	12	1	4887.05	4887.05	60	3509.37	5.49
7	5	15	1	na	na	60	1437.34	9.08
8	5	12	2	na	1733.47	60	1067.07	2.74
9	5	13	2	4630.17	4630.17	60	1339.71	5.04
10	5	14	2	na	na	60	2321.07	7.48

5.4.3. Nonlinear solver benchmark

As a third benchmark, COIN-OR Couenne (Convex Over and Under Envelopes for Nonlinear Estimation) is used to solve the DRLP test cases. Couenne is an open-source solver that can handle mixed-integer nonlinear programming (MINLP) problems. In our test runs, we used the GAMS modeling interface to access the Couenne solver. The nonlinear DRLP problem is coded in GAMS and then sent to the solver. GAMS is a sophisticated modeling tool designed for modeling linear, nonlinear and mixed integer optimization tools. One advantage of GAMS is that third-party optimization solvers are easily integrated into this interface. For our test cases, we used GAMS (version 23.7.3) installed on hardware with i5-7200 CPU and 8 GB RAM.

For the test cases, we started with the smaller instances. As seen in Table 10, convergence was established in only two cases when Couenne was not provided with an initial solution. Even in those cases, the performance in terms of total flow cost (i.e., the objective function) is worse than that of the proposed approach. To facilitate convergence, we introduced a partial initial solution where we setup binary variables to an arbitrary solution. Deploying a partial initial solution helps some non-convergent cases to converge, but the initialization does not improve the quality of the solution if the case is already convergent. In the test cases, the run time is capped at 60 min.

We also explored sub-problems that can be solved by nonlinear solvers. Considering that the DRLP problem at hand includes flow splitting, non-zero corridor width and replicate machines, a bare-minimum problem with such features is still a challenging problem for nonlinear solvers. In our search for a sub-problem, we noticed that we have to relax some of features (e.g., no machine replication or zero corridor width) so that we can have full convergence.

5.5. Impact of flow-splitting

One of the unique aspects of this study is the use replicate machines. This is a practical option that has been all but ignored in the previous literature. Using replicates allows flow to split into multiple branches where each branch represents a parallel processing line. To illustrate the benefit of splitting, we used a test case to observe the relationship between flow cost and number of replicates in the layout. The test case includes 10 different machine types, and two products with 16 and 18 machining processes, respectively. One of the machine types (type #8) is designated as a critical asset because of the high number of visits by both products. The machining sequence for both products are given in Table 11. While the first product visits machine type #8 three times during the manufacturing process, the second product requires the same machine type five times.

In the initial layout we use just one replicate of machine type #8. The capacity of the machine is set to a level that is equal to the total flow requirement (i.e., $3 \times 101 + 5 \times 435 = 2478$). This leaves zero slack capacity in the machine and the production lines of both products have to use this single machine. The total flow cost of the congested initial layout is used as a basis for comparison. As we introduced second and third machines (each with a capacity of 2478 units) to the layout, the cost starts to decline. Fig. 5 shows the total flow cost with respect to number of replicates for machine type #8. As this graph shows, the most significant cost improvement happens when a second machine of type #8 is added to the layout. The improvement gradually declines with increasing number of replicates. The total flow cost on the Figures is based on the average of 10 random runs. To eliminate the sensitivity of final numbers to initial random seed, the problem is solved 10 times with different starting seed at each level of number of replicates.

The capability to split flow into multiple branches improves the total flow cost significantly. This is particularly true if there are reentrant flows in the network (i.e., products visit the same machine type more than once). In the preceding example we assume identical replicates to increase capacity. Some of that capacity may be redundant. If we use replicates with different capacity and physical size (i.e., smaller width) there could be further improvement in both total flow cost and investment cost.

Table 11
Production sequence.

Product	Production quantity	Number of machines	Machining sequence
1	101	16	4, 6, 2, **8**, 10, 6, 4, 9, 4, **8**, 10, 2, 5, **8**, 6, 7
2	435	18	6, 3, **8**, 2, 6, 1, **8**, 10, 9, 1, 3, 4, 9, **8**, 2, **8**, 3, **8**

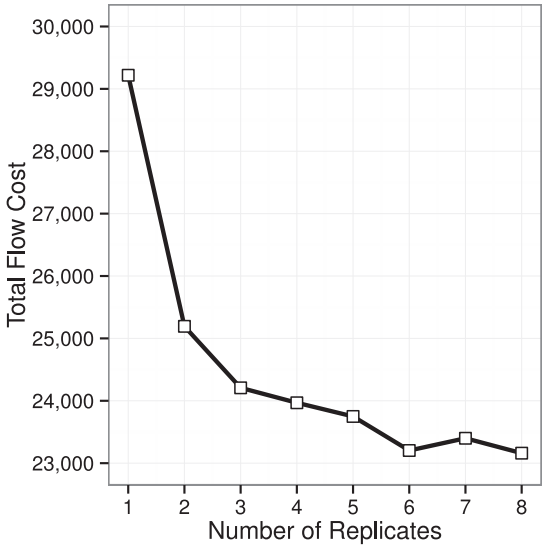


Fig. 5. Total flow cost vs. number of replicates of type #8.

6. Summary and future research directions

This paper defines a more realistic and complex version of the double row layout problem that incorporates classes of capacitated machines. Although the machines in a particular class perform the same function, these machines may differ by their size or processing rates. This problem considers multiple machine classes and distinct process flows for multiple products. The placement of machines as well as the optimal splits of product flow among machines are made simultaneously. This problem is most appropriate for greenfield sites, where there are no costs associated with modifying an existing layout and there are no physical restrictions regarding the dimensions of the overall layout. However, opportunities exist to extend this application to reconfiguration of existing facilities.

A nonlinear mixed integer programming formulation for this problem was shown. Due to the difficulties associated with exactly solving such a problem, we developed and tested a hierarchical heuristic for determining machine placements and corresponding material flows. While there are no known approaches for determining provably optimal solutions to problems of practical size, our numerical analysis indicates that the heuristic performs very well for all problem sizes. Computation time for a given number of machines increases roughly linearly with increases in product types and production sequences. Our computational experience shows the heuristic approach is robust and yields high quality solutions. Our work also shows the advantage of allowing split flows in reducing total flow (material handling) cost. There are also likely benefits in improving throughput and overall machine utilization with such an approach.

Numerous opportunities for future research exist. For example, we have considered a single period where the production quantity for each product type remains constant. Extensions that consider multiple time periods would be of value. Thus, if each product type has a planned production quantity over time, layouts could be generated that incorporate these demand fluctuations. Furthermore, as demand changes, tool installation/removal considerations become important, whereby objectives of minimizing cost and disruptions to the layout would be critical. Finally, future research may consider extending this work beyond the DRLP to address more than two rows.

References

Amaral, A., 2006. On the exact solution of a facility layout problem. Eur. J. Oper. Res. 173 (2), 508–518.

Amaral, A., 2008. An exact approach to the one-dimensional facility layout problem. Oper. Res. 56 (4), 1026–1033.

Amaral, A., 2011. On duplex arrangement of vertices. Technical Report. Departamento de Informatica, Universidade Federal do Espirito Santo (UFES), Brazil.

Amaral, A., 2012. The corridor location problem. Comput. Oper. Res. 39, 3325–3330.

Amaral, A., 2013. Optimal solutions for the double row layout problem. Optim. Lett. 7 (2), 407–413.

Amaral, A., 2013. A parallel ordering problem in facilities layout. Comput. Oper. Res. 40 (12), 2930–2939.

Amaral, A., 2018. A mixed-integer programming formulation for the double row layout of machines in manufacturing systems. Int. J. Prod. Res. doi:10.1080/00207543.2018.1457811. In press

Audet, C., Brimberg, J., Hansen, P., Le Digabel, S., Mladenovic, N., 2004. Pooling problem: alternate formulations and solution methods. Manage. Sci. 50 (6), 761–776.

Bureau, M., Dauzere-Peres, S., Mati, Y., 2006. Scheduling challenges and approaches in semiconductor manufacturing. In: IFAC Proceedings Volumes. Elsevier, pp. 739–744.

Campbell, A., Savelsbergh, M., 2004. A decomposition approach for the inventory-routing problem. Transp. Sci. 38 (4), 488–502.

Castillo, I., Peters, B., 2003. An extended distance-based facility layout problem. Int. J. Prod. Res. 41 (11), 2451–2479.

Castillo, I., Peters, B., 2004. Integrating design and production planning considerations in multi-bay manufacturing facility layout. Eur. J. Oper. Res. 157 (3), 671–687.

Chung, J., Tanchoco, J., 2010. The double row layout problem. Int. J. Prod. Res. 48 (3), 709–727.

Erbeyoglu, G., Bilge, U., 2016. PSO-based and SA-based metaheuristics for bilinear programming problems: an application to the pooling problem. J. Heuristics 22 (2), 147–179.

Gupte, A., Shabbirand, D., Santanu, S., Cheon, M., 2016. Relaxations and discretizations for the pooling problem. J. Global Optim. 1–39.

Harjunkoski, I., Grossmann, I., 2001. A decomposition approach for the scheduling of a steel plant production. Comput. Chem. Eng. 25 (11–12), 1647–1660.

Heragu, S., Kusiak, A., 1988. Machine layout problem in flexible manufacturing systems. Oper. Res. 36 (2), 258–268.

Heragu, S., Kusiak, A., 1991. Efficient models for the facility layout problem. Eur. J. Oper. Res. 53 (1), 1–13.

Ioannou, G., 2007. An integrated model and a decomposition-based approach for concurrent layout and material handling system design.. Comput. Ind. Eng. 52 (4), 459–485.

Jaramillo, J., McKendall Jr., A., 2010. The generalised machine layout problem. Int. J. Prod. Res. 48 (16), 4845–4859.

Kosucuoglu, D., Bilge, U., 2012. Material handling considerations in the FMS loading problem with full routing flexibility. Int. J. Prod. Res. 50 (22), 6530–6552.

Kulturel-Konak, S., 2012. A linear programming embedded probabilistic tabu search for the unequal-area facility layout problem with flexible bays. Eur. J. Oper. Res. 223 (3), 614–625.

McKendall, A., Jaramillo, J., 2006. A tabu search heuristic for the dynamic space allocation problem. Comput. Oper. Res. 33 (3), 768–789.

McKendall Jr., A.R., 2008. Improved tabu search heuristics for the dynamic space allocation problem. Comput. Oper. Res. 35 (10), 3347–3359.

Montreuil, B., Nourelfath, M., Brotherton, E., 2004. Coupling zone-based layout optimization, ant colony system and domain knowledge. In: Progress in material handling research. Material Handling Institute of America, Charlotte, pp. 301–333.

Murray, C., Zuo, X., Smith, A., 2012. An extended double row layout problem. 12th IMHRC International Material Handling Research Colloquium. Gardanne, France.

Murray, C.C., Smith, A.E., Zhang, Z., 2013. An efficient local search heuristic for the double row layout problem with asymmetric material flow. Int. J. Prod. Res. 51 (20), 6129–6139.

Nahapetyan, A., Pardalos, P., 2007. A bilinear relaxation based algorithm for concave piecewise linear network flow problems. J. Ind. Manage. Optim. 3 (1), 71–85.

Romero, R., Monticelli, A., 1994. A hierarchical decomposition approach for transmission network expansion planning. IEE Trans. Power Syst. 9 (1), 373–379.

Samarghandi, H., Eshghi, K., 2010. An efficient tabu algorithm for the single row facility layout problem. Eur. J. Oper. Res. 205 (1), 98–105.

Secchin, L., Amaral, A., 2018. An improved mixed-integer programming model for the double row layout of facilities. Optim. Lett. doi:10.1007/s11590-018-1263-9. In press

da Silva, G.C., Bahiense, L., Ochi, L.S., Boaventura-Netto, P.O., 2012. The dynamic space allocation problem: applying hybrid GRASP and Tabu search metaheuristics. Comput. Oper. Res. 39 (3), 671–677.

Solimanpur, M., Jafari, A., 2008. Optimal solution for the two-dimensional facility layout problem using a branch-and-bound algorithm. Comput. Ind. Eng. 55 (3), 606–619.

Solimanpur, M., Kamran, M., 2010. Solving facilities location problem in the presence of alternative processing routes using a genetic algorithm. Comput. Ind. Eng. 59 (4), 830–839.

Taghavi, A., Murat, A., 2011. A heuristic procedure for the integrated facility layout design and flow assignment problem. Comput. Ind. Eng. 61 (1), 55–63.

Tubaileh, A., Siam, J., 2017. Single and multi-row layout design for flexible manufacturing systems. Int. J. Comput. Integr. Manuf. 30 (12), 1316–1330.

Turley, J., 2002. The Essential Guide to Semiconductors. Prentice Hall.

Urban, T., Chiang, W., Russell, R., 2000. The integrated machine allocation and layout problem. Int. J. Prod. Res. 38 (13), 2911–2930.

Wang, S., Zuo, X., Liu, X., Zhao, X., Li, J., 2015. Solving dynamic double row layout problem via combining simulated annealing and mathematical programming. Appl. Soft Comput. 37, 303–310.

Yao, W., Zhao, J., Wen, F., Xue, Y., Ledwich, G., 2013. A hierarchical decomposition approach for coordinated dispatch of plug-in electric vehicles. IEEE Trans. Power Syst. 28 (3), 2768–2778.

Zhang, M., Batta, R., Nagi, R., 2011. Designing manufacturing facility layouts to mitigate congestion. IIE Trans. 43 (10), 689–702.

Zhang, Z., Murray, C., 2012. A corrected formulation for the double row layout problem. Int. J. Prod. Res. 15 (50), 4220–4223.

Zhao, Y., Wallace, S., 2013. Facility layout design with random demand and capacitated machines. Working Paper. http://eprints.lancs.ac.uk/61809/1/paper_01_WP.pdf

Zuo, X., Murray, C., Smith, A., 2014. Solving an extended double row layout problem using multi-objective tabu search and linear programming. IEEE Trans. Autom. Sci. Eng. 11 (4), 1122–1132.

Zuo, X., Murray, C.C., Smith, A.E., 2016. Sharing clearances to improve machine layout. Int. J. Prod. Res. 54 (14), 4272–4285.

Zuo, X., Murray, C.C., Smith, A.E., 2016. The double-bay layout problem. IEEE Trans. Semicond. Manuf. 29 (4), 446–454.