An Improvement-type Layout Algorithm for Single and Multiple-floor Facilities

Yavuz A. Bozer • Russell D. Meller • Steven J. Erlebacher
Department of Industrial and Operations Engineering, University of Michigan, Ann Arbor, Michigan 48109-2117
Department of Industrial Engineering, Auburn University, Auburn, Alabama 36849-5346
John M. Olin School of Business, Washington University, St. Louis, Missouri 63130

In this paper we introduce the use of spacefilling curves in facility layout, and we extend a well-known facility layout algorithm (CRAFT) to facilities with multiple floors. Spacefilling curves make it possible to exchange any two departments and to use more powerful exchange routines than two-way or three-way exchanges. We also further enhance CRAFT by controlling department shapes, and (with multiple floors) by allowing "flexible" departmental area requirements. Although the algorithm we present can be used for any single-floor or multi-floor facility layout problem, its primary target is production facilities. A tailored version of the algorithm was successfully tested and used in a large, multi-floor production facility. The algorithm differs significantly from two previous extensions of CRAFT to multi-floor facilities.
(*Facility Layout; Improvement Heuristics; Multi-floor Facilities; Layout Algorithms*)

1. Introduction

According to Tompkins and White (1984), the "generation of layout alternatives is a critical step in the facilities planning process." Given certain interactions that occur among the departments, generally speaking, the facility layout problem is concerned with determining the "most efficient" arrangement of the departments subject to constraints imposed by the site plan, the building, the departmental area/service requirements, and the decision-maker.

Obtaining optimal solutions to the facility layout problem is not straightforward primarily for two reasons. First, there are no generally accepted objective functions which capture all the relevant aspects of the problem. Second, with commonly used objective functions, finding the optimal solution is currently near-impossible since it often leads either to a large-scale Quadratic Assignment Problem (QAP) or a large-scale mixed integer programming problem (Montreuil 1990). Thus, most of the research has been aimed at developing heuristic procedures. Generally speaking, the multi-floor layout problem is more complicated than its single-floor counterpart since the former involves vertical flow and lifts. Also, in a multi-floor problem, the number of departments that can be assigned to any floor is limited by available space on that floor. Consequently, some layouts may not be area feasible. Of course, there may be additional constraints imposed by a multi-floor building. We discuss some of these constraints in §4.2.

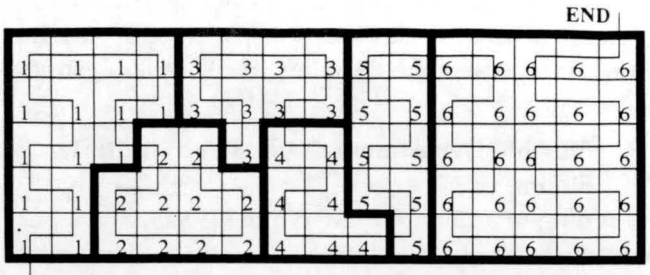
Most new production facilities are single-story buildings. However, multi-floor production facilities are still in use in the United States, and some new production facilities are constructed with multiple floors; see, for example, *Industrial Engineering* (1990a), where the company "saved on land and construction costs by going up three floors." Also, according to *Material Handling Engineering* (1988a) ". . . one major (factor) which recommends renovation is its low cost compared to that of a new building . . . (and) . . . most old buildings are multi-story." In fact, "a refurbished old industrial plant . . . is likely to cost as little as a tenth as much per square foot of building as a new factory today" (King and Johnson 1983). We are not arguing that renovating a multi-floor plant is always the preferred alternative. Rather, we are stressing that there are many multi-floor industrial buildings that have been

0025-1909/94/4007/0918$01.25

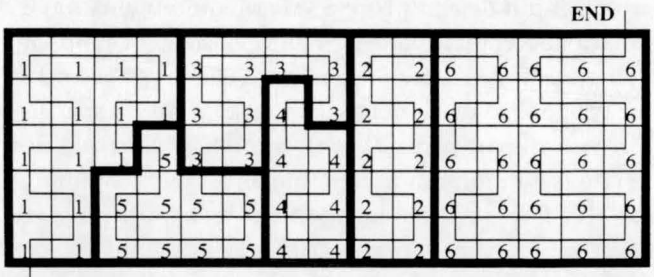
visiting other grids. As we demonstrate later, using spacefilling curves offers significant advantages.

A six-department example is shown in Figure 1*a*. With a given spacefilling curve and department area values, a layout is uniquely determined only by a sequence of department numbers. Suppose departments 1 through 6 require 15, 10, 9, 7, 9 and 25 grids, respectively. Then, a possible layout is shown in Figure 1*a* where the *layout sequence* is given by 1–2–3–4–5–6. The layout was constructed by assigning the first 15 grids visited by the curve to department 1, the next 10 grids to department 2, and so on. Unlike CRAFT, spacefilling curves allow MULTIPLE to exchange any two departments whether or not they are adjacent and/or equal in area. For example, in Figure 1*a*, to exchange departments 2 and 5, we simply exchange their positions in the layout sequence and obtain the new layout shown in Figure 1*b*. Note that the locations of departments 3 and 4 have shifted "automatically" since they fall between departments 2 and 5.

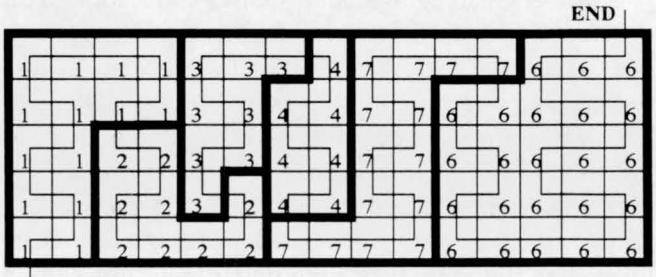
Figure 1 Using Spacefilling Curves to Construct Layouts



(a) Initial Layout.



(b) Department 2 and Department 5 Exchanged.



(c) Department 5 and Department 7 Exchanged.

3.2. Generating Spacefilling Curves

In a rectangular building with no fixed departments or unusable areas, the spacefilling curve can be obtained by using a recursive procedure (see the Hilbert curve in the Appendix). If the building shape is irregular, and many obstacles and/or fixed departments are present, one may generate a spacefilling curve by hand. Although such a curve may mathematically not be a spacefilling curve, *functionally* it serves as one. Using MULTIPLE in a large, four-floor production facility, for example, we opted for hand-generated curves. In doing so, we fully captured the exact building shape, the current layout, and all the obstacles. Any curve (hand-generated or otherwise) which visits all the grids by taking horizontal, vertical, or diagonal steps (from one grid to an adjoining grid) can be used with MULTIPLE. Such alternate curves can be generated interactively on a personal computer. We have also generated some curves by solving a TSP, where one can minimize the number of diagonal steps. Note that spacefilling or alternate curves are generated only once for each floor at the start.

A spacefilling curve may be tailored to a particular building. If a department is fixed, the curve simply bypasses all the grids assigned to that department. Otherwise, even if we disallow every exchange that involves a fixed department, it may still shift. In reference to Figure 1a, if department 2 is fixed, we simply reroute the curve so that none of the grids assigned to department 2 are visited. However, if department 5 is fixed, we would generate two spacefilling curves, one for each section of the floor. The initial layout may also affect curve generation. Curves that fully conform to the initial layout can be generated by solving a *clustered* TSP (Chisman 1975). Lastly, it is instructive to note that the *sweep method* used by ALDEP (Seehof and Evans 1967) can be viewed as a "spacefilling curve." Although the

sweep method also can use a layout sequence to rapidly construct a layout, it may split departments around fixed departments or obstacles. The sweep method is too rigid; it does not possess the flexibility of spacefilling curves.

3.3. Flexible Departmental Areas and Multiple Floors

According to Tompkins and White (1984), ". . . perhaps the most difficult determination in facilities planning is the amount of space required in the facility." Similarly, Lew and Brown (1968) state that "in any architectural design process, area requirements have a range of acceptable values." Hence, instead of supplying a single estimate for the departmental area requirements, we propose to use a *range of acceptable values* specified for each department. That is, we let A_i^L and A_i^U designate the minimum acceptable and the maximum allowable floor space to be allocated to department i, respectively. The area of department i in the initial / current layout, say, A_i, is assumed to fall within its corresponding range.

Consider exchanging departments i and j, located on different floors. Without loss of generality, suppose $A_i > A_j$. If $A_i^L \le A_j$ and $A_j^U \ge A_i$, then the exchange is area feasible and it represents an *even exchange*; i.e., the two departments can be exchanged without having to relayout either floor. If the above two conditions are not met, one might still be able to exchange departments i and j after "compressing" the departments currently located on the same floor with department j by setting their areas equal to their lower limits. (We need not compress the departments currently located on the same floor with department i since $A_i > A_j$.) If department i fits in the space that becomes available, then the exchange is area feasible. Otherwise, departments i and j cannot be exchanged in the current layout. Following an exchange, any additional floor space is left unused at the end of the spacefilling curve. The above procedure is formally presented in Figure 2, where $k(j)$ denotes the floor number of department j, $S_{k(j)}$ denotes the set of departments located on floor $k(j)$, and $T_{k(j)}$ denotes the total area available on floor $k(j)$.

Consider the previous example shown in Figure 1a. Suppose the lower, current, and upper limits on the area for each of the six departments are given as follows:

$$12 \le A_1\,(=15) \le 17, \quad 8 \le A_2\,(=10) \le 12,$$

$$7 \le A_3\,(=9) \le 12,$$

$$6 \le A_4\,(=7) \le 10, \quad 6 \le A_5\,(=9) \le 12,$$

$$20 \le A_6\,(=25) \le 30.$$

Consider next exchanging departments 5 and 7, where the latter is currently located on a different floor and $12 \le A_7\,(=14) \le 17$. Since $A_7^L > A_5$, an even exchange is not possible. However, with compression, the exchange is area feasible. Thus, following the procedure shown in Figure 2, we ultimately obtain the layout shown in Figure 1c. Without spacefilling curves, compression would not be possible since some compressed departments may have to shift to accommodate the entering department.

3.4. Controlling Department Shapes

As a department becomes irregular in shape, it becomes more difficult to develop an efficient *detailed layout* for that department. With MULTIPLE, following an exchange, some department shapes may not be acceptable. (The same problem has also been reported for CRAFT; see Hicks and Cowen (1976) and Lew and Brown (1968), among others.) While the human eye is very adept at making judgments concerning shape, a computer program requires a formal measure, which must also be easy to compute. Freeman (1974) notes that, for a fixed area, the perimeter of an object increases as it becomes more irregular in shape. Letting P_i denote the perimeter of department i, one can use P_i / A_i as a measure of shape irregularity. (With the matrix representation, P_i can be computed with relative ease.)

The perimeter of a (non-circular) object would be minimized if the object is square shaped. Therefore, the minimum perimeter for department i, say, P_i^*, is equal to $4\sqrt{A_i}$. Assuming that a square represents the ideal department shape, the normalized shape measure for department i, say, Ω_i, is given by:

$$\Omega_i = \frac{P_i / A_i}{P_i^* / A_i} = \frac{P_i}{P_i^*} = \frac{P_i}{4\sqrt{A_i}} = \frac{1}{4} P_i A_i^{-0.5}. \qquad (1)$$

With the above measure, as the department shape becomes more irregular, its Ω_i value increases. Hence, for each department, the analyst specifies an upper limit on Ω_i. Our experience suggests that reasonable shapes

(or will be) renovated. Coupled with recent advances in vertical material handling technology (see, for example, *Industrial Engineering* 1990a, 1990b; *Material Handling Engineering* 1988b, 1990; *Modern Materials Handling* 1985), many firms are likely to consider renovating or constructing multi-floor buildings, particularly in those cases where land is limited.

In this paper we extend CRAFT (Armour and Buffa 1963, Buffa et al. 1964) by applying spacefilling curves to single or multi-floor facility layout problems. Such curves generally increase the number of department exchanges considered at each iteration. We also add new features such as shape control for departments. Furthermore, for multi-floor facilities, we allow flexible departmental area requirements and the consideration of additional constraints which are described later. The remainder of the paper is organized as follows. In §2 we present the literature review. In §3 and 4 we present our layout improvement algorithm. In §5 and 6 we evaluate the new algorithm in a multi-floor and a single-floor setting, respectively. In §7 we present certain extensions to the algorithm. Lastly, in §8 we present our conclusions and describe the implementation of the algorithm in a real-life multi-floor production facility.

2. Literature Review

A number of computer-based heuristic layout algorithms, such as ALDEP (Seehof and Evans 1967), BLOCPLAN (Donaghey and Pire 1990), COFAD (Tompkins and Reed 1976), CRAFT (Armour and Buffa 1963, Buffa et al. 1964), and SHAPE (Hassan et al. 1989) have been developed over the years. There are also a number of algorithms based on graph theory (see Drezner 1980, Foulds 1983, and Foulds and Robinson 1978, among others), where each department is initially represented as a node. After a planar graph is developed to identify adjacent departments, a heuristic procedure is applied to construct a block layout (see, for example, Montreuil et al. 1987).

CRAFT—Computerized Relative Allocation of Facilities Technique—is a well-known improvement-type layout algorithm. Despite some of its shortcomings, CRAFT represents the cornerstone of improvement algorithms. It has been shown to give reasonably good solutions to a variety of (single-story) layout problems (Ritzman 1972). Also, CRAFT can capture in reasonable detail the building shape (including unusable areas) and the current layout, including fixed departments. CRAFT begins with an initial layout and performs two-way and/or three-way exchanges of *department centroids* to identify those that potentially reduce the layout cost, which is based on (flow) $\times$ (unit cost) $\times$ (rectilinear distance between department centroids). At each iteration, the exchange that leads to the largest *estimated* reduction in total cost is selected. The improvement process is then restarted with the new layout and it continues until there are no 2-way or 3-way exchanges that reduce the estimated layout cost. Like most steepest–descent algorithms, CRAFT is a "path dependent" heuristic; i.e., the initial layout and the exchanges considered at each iteration largely determine the quality of the final solution.

It is well-known that CRAFT can exchange only those departments that are either adjacent or equal in area. If two non-adjacent departments (with unequal areas) are exchanged, other departments must be "shifted"; otherwise, one of the departments being exchanged will be "split." CRAFT is not capable of shifting the other departments, and splitting a department is not acceptable in a production facility. Also, in most real-world layout problems, only a few, if any, of the departments will have exactly the same area requirements. Consequently, the above constraint significantly reduces the number of exchanges CRAFT would consider at each iteration. Combined with path dependency, this is likely to adversely affect the cost of the final solutions obtained by CRAFT.

The remainder of the literature review is limited to multi-floor layout algorithms. SPACECRAFT (Johnson 1982) is similar to CRAFT except that vertical travel is nonlinear, and the facility is "transformed" into a single floor to identify department exchanges. The transformation is performed (at each iteration) by "appending" each floor, one at a time, to the first floor. Potential exchanges are then identified as in CRAFT; i.e., each potential exchange is performed on the single-floor layout, which is then separated back into multiple floors to evaluate the objective. In the separation process, a department may be split across two or more floors. Although this may be acceptable for some office buildings, in a production facility a department is an indivisible entity, by definition. Also, it is not clear how one would

allocate the incoming and outgoing flow among the pieces of a split department. If a department is split across floors, note that it also creates additional *vertical handling within the department*, which is not captured in the objective. Lastly, SPACECRAFT models nonlinear vertical travel in the objective by assigning an expected waiting time to each lift. The user enters the expected waiting times *a priori* for each lift.

A similar adaptation of CRAFT to multiple floor facilities, CRAFT-3D, is presented by Çınar (1975). The details of CRAFT-3D are not explained in Çınar (1975) and we have not located a refereed paper by the author on multi-floor layout. Nevertheless, SPACECRAFT and CRAFT-3D appear to be similar. For details, the reader may refer to Jacobs (1984). We stress that both SPACECRAFT and CRAFT-3D become identical to CRAFT if they are run with the number of floors set equal to one.

Other layout algorithms developed for multi-floor facilities are of the *construction type*; i.e., they construct a layout in an empty building. Since such layout algorithms are beyond the scope of our paper, we will only name them: Automated Layout Design Program (ALDEP) by Seehof and Evans (1967), Space Planning Systems by Liggett and Mitchell (1981), Multi-Story Layout Program by Kaku et al. (1988), and BLOCPLAN by Donaghey and Pire (1990). Typically, these algorithms use heuristic procedures to first assign the departments to floors, and then to layout each floor. They have one or more of the following limitations: lift locations are either not considered or only one centrally located bank of lifts is allowed, all departments are equal in area, and/or the layout of each floor is determined independently. Although ALDEP may be indirectly used as an improvement algorithm, its use in a multi-floor setting is unclear. Neither the original paper nor subsequent publications address the topic in detail. In fact, we do not know how the departments are assigned to floors. Once the assignment is made, however, ALDEP uses the "sweep method" to layout each floor independently of the others. Interactions that occur between departments on different floors are ignored (Çınar 1975, p. 25).

In short, although a few improvement or construction-type layout algorithms are available for multi-floor facilities, certain factors would limit their use in a multi-floor production facility. Except for the throughput capacity of the lifts, the improvement-type algorithm we present here overcomes these limitations and seems to generate reasonably good solutions for multiple as well as single-floor facilities. Before we present the algorithm, in the next section we present the use of spacefilling curves, flexible areas, and departmental shape constraints in facility layout. These concepts and their integration into one algorithm represent our main contribution.

3. Spacefilling Curves, Flexible Areas, and Shape Control

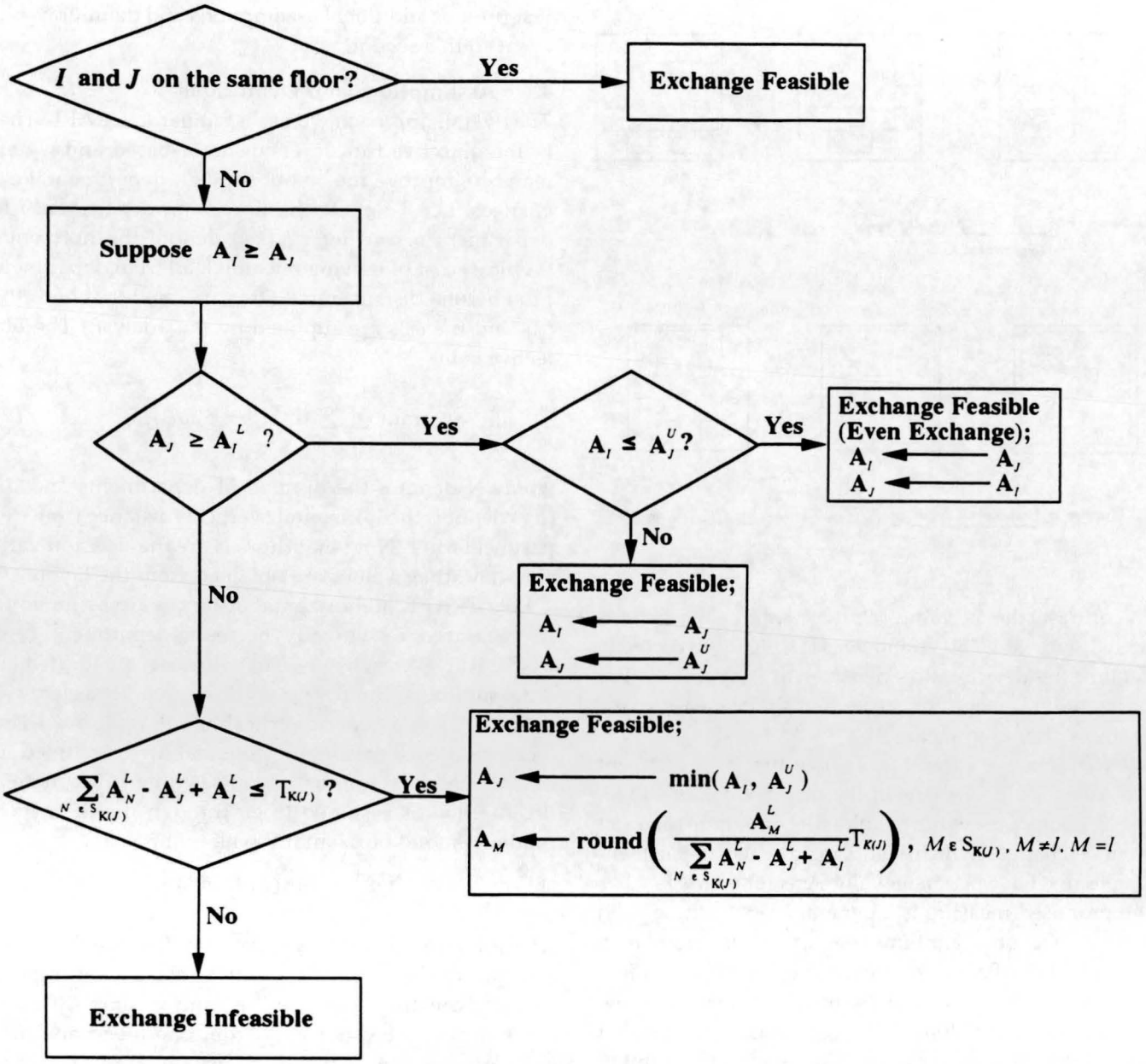
In this section we describe basic concepts which lead to the development of the new improvement algorithm, namely, **MULTIPLE** (*MULTI*-floor *P*lant *L*ayout *E*valuation). After discussing layout representation with spacefilling curves, we illustrate how spacefilling curves and flexible department areas allow MULTIPLE to increase the number of exchanges within a floor and the number of exchanges across floors (without splitting departments). We also show how MULTIPLE controls department shapes.

3.1. Spacefilling Curves and the Facility Layout Problem

Spacefilling curves, which have long been studied by mathematicians, have been recently proposed as a Traveling Salesman Problem (TSP) heuristic by Bartholdi and Platzman (1982), who also used such curves to locate items in a storage rack (Bartholdi and Platzman 1988).

As in most computer-based layout algorithms, in MULTIPLE the layout is represented as a matrix. Each element of the matrix corresponds to a grid square (or grid) of specified area, and the space required by each department is expressed as an integer number of grids. To construct the layout, we propose to use a spacefilling curve which simply visits all the grids on a floor. To ensure that a department is not split, all the grids assigned to a department must be contiguous, i.e., each grid must be adjacent to another grid that has been assigned to the same department. A spacefilling curve can guarantee that no departments will be split because a separate curve is used for each floor and, within each floor, the curve visits the "neighbors" of a grid before

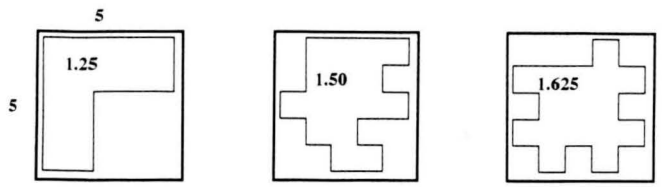
Figure 2 **Procedure to Determine Area Feasibility and Department Areas**



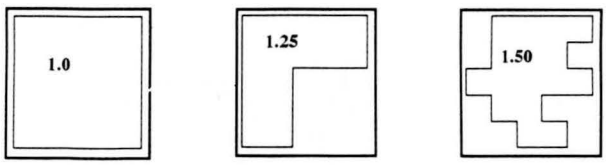
are generally obtained if this upper limit is kept under 1.50. If the ideal shape for a department is not a square, one needs to redefine the minimum perimeter and impose a lower limit on Ω_i as well.

Two alternative shape measures were proposed by Liggett and Mitchell (1981). The first one is obtained by dividing the area of the smallest enclosing rectangle by the area of the department itself. The second measure is obtained by dividing the length of the smallest enclosing rectangle by its width. As shown in Figure 3, neither measure is as effective as Ω_i. Consider first Figure 3a, where the department area is fixed at 16 grid squares and the ideal shape is a square. Using the first shape measure, we obtain 25 / 16 = 1.5625 for *all three shapes*.

Figure 3 Comparison with Liggett and Mitchell Shape Factors



(a) Measure 1: $\frac{\text{Encl. Rect. Area}}{\text{Department Area}} = 25/16 = 1.5625$



(b) Measure 2: $\frac{\text{Encl. Rect. Length}}{\text{Encl. Rect. Width}} = 1.0$

In contrast, the Ω_i value (shown within each department) starts at 1.250 and increases to 1.625 to correctly capture the deteriorating department shape. A similar observation is made for the second shape measure; see example shown in Figure 3b.

Of course, we cannot guarantee that a department will attain its ideal shape in the final layout. However, the proposed measure enables the program to reject some exchanges which result in irregular department shapes that have perimeter values greater than P_i^*. Also, the proposed measure is a general one; it can be used with most layout algorithms. The spacefilling curve may also affect the final department shapes. Although it is not possible to predict department shapes from the spacefilling curve alone, with such curves department shapes do not necessarily deteriorate with the number of iterations. (The department shapes in CRAFT have a tendency to deteriorate fairly rapidly.)

4. The Layout Improvement Algorithm

The new layout algorithm we developed (MULTIPLE) is presented in this section. The algorithm is based on integrating the concepts described in §3 within the framework of a steepest–descent search heuristic. We first present additional assumptions and definitions relevant to this section.

4.1. Assumptions and Definitions

The overall approach we use is similar to CRAFT. That is, the objective function is distance-based and we attempt to improve the layout through departmental exchanges. Let f_{ij} denote the flow from department i to department j, and let c_{ij}^H (c_{ij}^V) denote the horizontal (vertical) cost of moving one unit load from department i to j by one distance unit. (It is assumed that both the f_{ij}'s and the c_{ij}'s are supplied by the analyst.) The objective is to

$$\min \sum_{i=1}^{N} \sum_{j=1}^{N} (c_{ij}^H d_{ij}^H + c_{ij}^V d_{ij}^V) f_{ij}, \tag{2}$$

where N denotes the number of departments and d_{ij}^H (d_{ij}^V) denotes the horizontal (vertical) distance from department i to j. Note that the d_{ij}'s are the decision variables and their values are obtained from the layout.

Like CRAFT, all horizontal distances are assumed to be measured rectilinearly between department centroids. However, when two departments are located on different floors, the flow goes through a lift. (Here we define a lift as a generic vertical handling device.) The location of each existing or potential lift is assumed to be specified in the initial layout. Letting l designate a lift, the flow is assumed to go through the lift which minimizes total horizontal travel; that is,

$$d_{ij}^H = \min_l (d_{il}^H + d_{lj}^H), \tag{3}$$

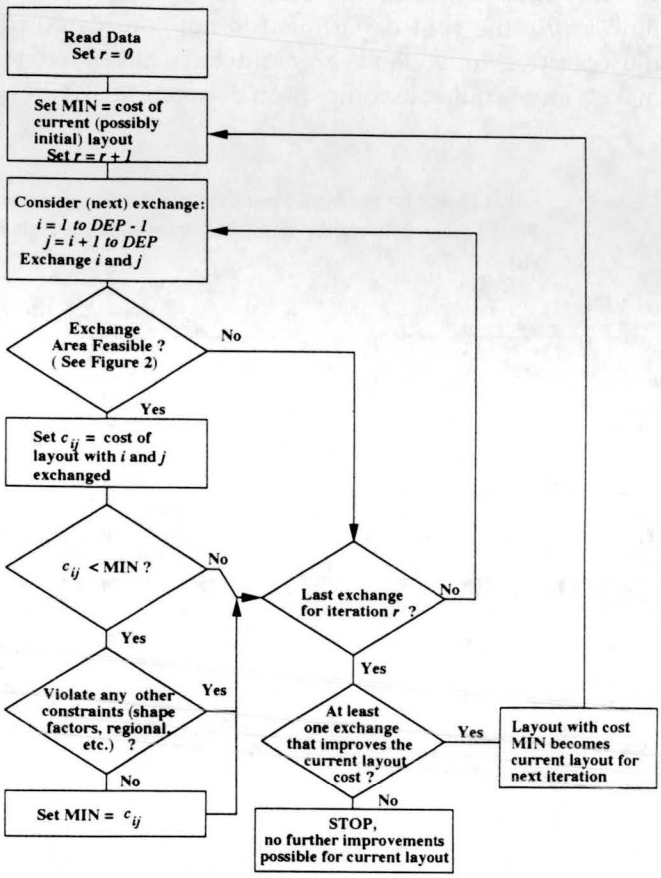
where d_{il}^H designates the horizontal distance from the centroid of department i to lift l. We do not impose capacity constraints on the lifts, and we implicitly assume that c_{ij}^V does not vary from one lift to another. (The latter assumption can be easily relaxed.) Furthermore, for each floor k, the analyst is assumed to supply available floor space, T_k (in grids), the initial layout sequence, and a spacefilling curve. The analyst must also supply A_i^L, A_i^U, and A_i (in grids) for $i = 1, 2, \ldots, N$.

4.2. Additional Constraints

In addition to the area and shape constraints, a department may be subject to floor constraints. That is, a department may have to be assigned to a particular floor,

or it cannot be located on certain floor(s). For example, the receiving/shipping department is (almost always) assigned to the ground floor. (One may also treat the receiving/shipping department as a fixed department.) Also, the floor loading capacity or the floor-to-ceiling distance may vary from one floor to another. Such floor constraints can be handled by disallowing an exchange if one or both of the departments involved violates a constraint as a result of that exchange. Another type of constraint is concerned with the particular section of a floor a department may not be assigned to. If such *regional constraints* are imposed on a department, the analyst simply indicates which grid squares these regions cover. After an exchange, if a department occupies a grid square that falls into its forbidden region, the exchange is disallowed. Accommodating the above constraints adds considerable realism to MULTIPLE. In the four-floor production facility where we applied MULTIPLE, we encountered all of the above constraints.

Figure 4 Flow Chart of MULTIPLE



4.3. The Algorithm

MULTIPLE is presented in Figure 4, where r is the iteration counter and *DEP* is the number of non-fixed departments. Starting with a given initial or current layout, at each iteration MULTIPLE considers all two-way, area-feasible exchanges between non-fixed departments. The impact on layout cost is measured according to equations (2) and (3). The current-best layout cost for each iteration is stored under *MIN*. When all exchanges have been evaluated, the algorithm selects the feasible exchange which yields the maximum reduction in the layout cost, and the exchange procedure is restarted with the new layout. The search procedure terminates when no feasible cost improving exchanges are identified in the current layout. Note that, at any given iteration, MULTIPLE considers all exchanges within each floor as well as all area-feasible exchanges across floors. In general, we believe this is a better strategy than a two-stage approach where one would first "optimize" by exchanging departments across floors, followed by exchanges within each floor.

5. Numeric Example for Multiple Floors

In this section we present an example to demonstrate MULTIPLE. The problem is based on a 15-department, three-floor facility with six existing (or potential) lifts. The departmental area data are shown in Table 1. (We assume there are no upper bounds imposed on department areas.) Department 15 (the receiving/shipping department) is fixed in its current location in the initial layout, which is shown in Figure 5a.

The spacefilling curve used for the second and third floors is shown in Figure 1a. (Note that the layout in Figure 1a corresponds to the third floor in Figure 5a.) The spacefilling curve for the first floor is identical to the other two except that it does not cover the last 25 grids (since department 15 is fixed). The flow matrix is shown in Table 2. For simplicity, the horizontal and vertical cost to travel a distance unit is assumed to be \$1.00 and \$5.00, respectively, for all pairs of departments—except the receiving/shipping department,

Table 1 Area Requirements for the Multi-Floor Example Problem

Department	Current Area	Minimum Area
1	15	12
2	10	7
3	9	6
4	7	5
5	9	7
6	25	22
7	25	22
8	15	13
9	10	7
10	25	22
11	10	9
12	15	13
13	6	4
14	19	17
15	25	25

which assumes a cost of \$0.25 (\$1.25) for horizontal (vertical) travel due to, say, larger unit loads. We assume 10 distance units between adjacent floors.

The initial layout cost is \$281,702.35. Using MULTIPLE (without department compression) on a 25 MHz DOS-based 386 personal computer with a 387 coprocessor (say, 386-PC), we obtained a final layout (shown in Figure 5b) with a cost of \$126,733.92, which represents a decrease of 55.01%. The solution was obtained in eight iterations, which took 20.2 seconds. We next explored the effect of department compression by solving the above problem with MULTIPLE's compression feature enabled. The final layout, obtained in seven iterations, has a cost of \$125,822.50. Hence, department compression accounts for an improved savings of only \$911.42. (The runtime with compression was 37.9 seconds. The increase is primarily due to evaluating a larger number of feasible exchanges at each iteration.) In the example problem, compression does not greatly improve the final layout cost. However, with larger and more realistic problems, compression may be essential.

Suppose we now impose a shape constraint on department 10 and assume that its ideal shape is a square. Setting the upper limit on Ω_{10} equal to 1.0 and rerunning MULTIPLE, we obtained the final layout shown in Figure 6a. (In general, an upper limit of 1.0 is too restrictive since near-square shapes will be rejected. This may severely limit the number of exchanges considered by the algorithm.) Clearly, for the example problem the shape constraint had little impact on the cost savings. The percentage reduction in cost decreased only from 55.01% to 54.32%. The final layout was still obtained in eight iterations (or 20.09 seconds) without compression.

For comparison purposes, we ran SPACECRAFT with the same data and in 12 iterations obtained the layout shown in Figure 6b. (We will not report the runtime since SPACECRAFT currently runs on a mainframe. Also, we set the expected waiting time at each lift equal to zero to force SPACECRAFT to use the closest lift.) In the final layout, SPACECRAFT split departments 8 and 10, and the cost is \$129,168.00. (The final layout cost obtained by MULTIPLE is \$125,822.50 and \$126,733.92, with and without compression, respectively.) Recall that, when a department is split, SPACECRAFT underestimates the layout cost since vertical flow *within* the split department is not considered in the objective function. We would have preferred to make a more equitable comparison between MULTIPLE

Figure 5 Initial Layout for the Multi-Floor Example Problem (a) and Final Layout Obtained by MULTIPLE without Compression (b)

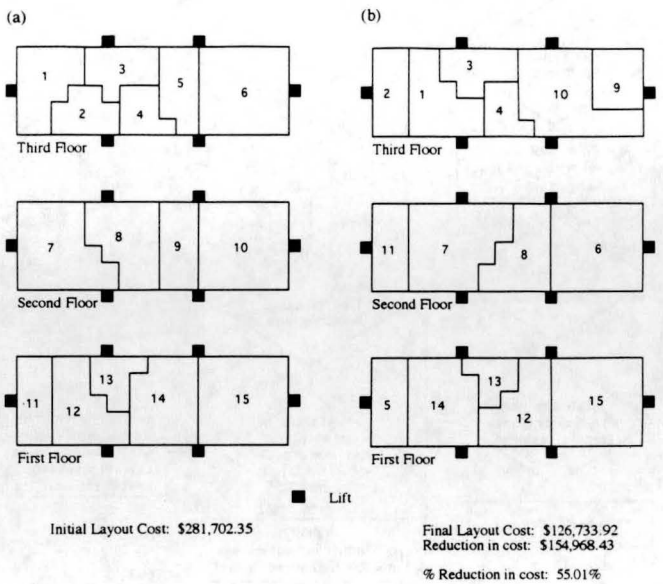


Table 2 Flow Matrix for the Multi-Floor Example Problem

From / To	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	240
2	240	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	0	0	0	1200	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	1200	0	0	0	0	0
5	0	0	0	0	0	0	0	0	0	0	0	0	0	600	0
6	0	0	0	0	0	0	0	480	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	480	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	120
9	0	0	0	0	0	0	0	0	0	600	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0	0	0	600	0	0	0
11	0	0	0	0	0	0	480	0	0	0	0	0	0	0	0
12	0	0	0	0	0	0	0	0	0	0	0	0	0	0	600
13	0	0	0	0	0	0	480	0	0	0	0	0	0	0	0
14	0	0	0	0	0	0	0	0	0	0	0	600	0	0	0
15	0	10	25	0	25	40	0	0	25	0	40	0	20	0	0

and SPACECRAFT. However, if one attempts to stop SPACECRAFT from splitting departments, then no two departments will be exchanged across floors unless they are equal in area. Also, since SPACECRAFT cannot handle flexible areas, we had to run it with fixed A_i values, which were obtained from the initial layout.

6. Computational Results for Single-Floor Problems

Although MULTIPLE is principally designed for multi-floor facilities, its use of spacefilling curves makes it very effective in single-floor layout problems as well. Since MULTIPLE can exchange any two departments, the set of exchanges considered by CRAFT at each iteration is a subset of those considered by MULTIPLE. However, since both algorithms are path dependent heuristics, one cannot guarantee that MULTIPLE will always outperform CRAFT. Rather, provided that both heuristics start with the same initial layout, in general, MULTIPLE is *likely to obtain a lower cost layout*. Thus, MULTIPLE represents a substantive improvement of CRAFT because it relaxes a major constraint.

In this section we compare MULTIPLE and CRAFT on single-floor problems. We first use a 20-department problem by Armour and Buffa (1963), where CRAFT was originally proposed. MULTIPLE's spacefilling curve (shown in Figure 7) fully conforms to the initial layout, which has a cost of \$101,643.37. In seven iterations CRAFT reduced the layout cost to \$78,620.90, whereas

Figure 6 Final Layout Obtained by MULTIPLE with Shape Constraint (a) and Final Layout Obtained by SPACECRAFT (b)

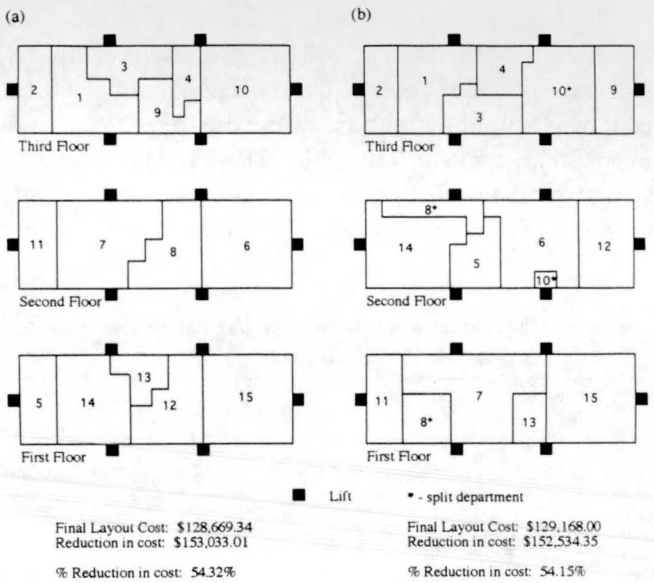
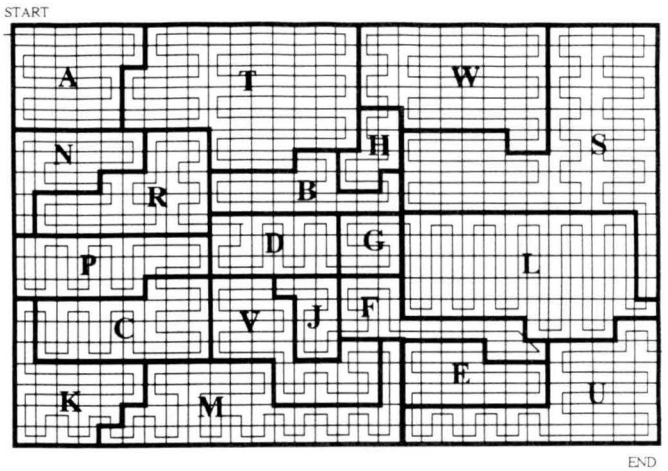


Figure 7 **Spacefilling Curve and Final Layout Obtained by MULTIPLE for the Example Problem Taken from Armour and Buffa (1963)**



MULTIPLE reduced it to \$68,578.83 in 13 iterations (see Figure 7 for the final layout). The shapes of departments F, M, and V are probably not acceptable. This is due, in part, to the conforming spacefilling curve. One may generate alternative spacefilling curves and / or impose shape constraints to improve the department shapes.

We next consider three additional data sets with 11, 15, and 25 departments. For problem data, the reader may refer to Meller (1992). Both MULTIPLE and CRAFT were run with 10 initial layouts for each data set. In 29 of the 30 problems, MULTIPLE produced a lower-cost layout than CRAFT. A summary of the results are shown in Table 3. On the average, MULTIPLE produced solutions that are approximately 11% lower in cost than those produced by CRAFT. The Wilcoxon signed-rank test—suggested by Golden and Stewart (1985) to compare heuristics—indicates that the differences shown in Table 3 are statistically significant (at 5%).

Table 3 **The Average and Standard Deviations of the Final Layout Cost for CRAFT and MULTIPLE**

Data Set	Craft		Multiple	
	Average	Standard Deviation	Average	Standard Deviation
11	1686.51	121.76	1526.90	124.45
15	41925.96	4119.12	38373.91	2231.06
25	2218.60	179.46	1970.71	146.30

By considering all possible layout sequences, it is possible to obtain the optimal layout *for a given spacefilling curve*. To assess the impact of alternate spacefilling curves, for the 11-department test problem we determined the "optimal" layout with three different spacefilling curves. The results are presented in Figure 8. (The curve shown in (a) was used in the above comparison with CRAFT.) The maximum "optimal" layout cost value, given by (a), is only 3.72% above the minimum, given by (c). Hence, for this problem, the choice of spacefilling curve has little impact on "optimal" layout cost. Further research is necessary to generalize such findings.

We ran MULTIPLE and CRAFT on the 386-PC described in §5. CRAFT solved the 20-department problem in 2.1 seconds, while MULTIPLE required 2.5 minutes. Other runtimes for the three test problems appear in

Figure 8 **Three Spacefilling Curves and the Resulting Optimal Layouts**

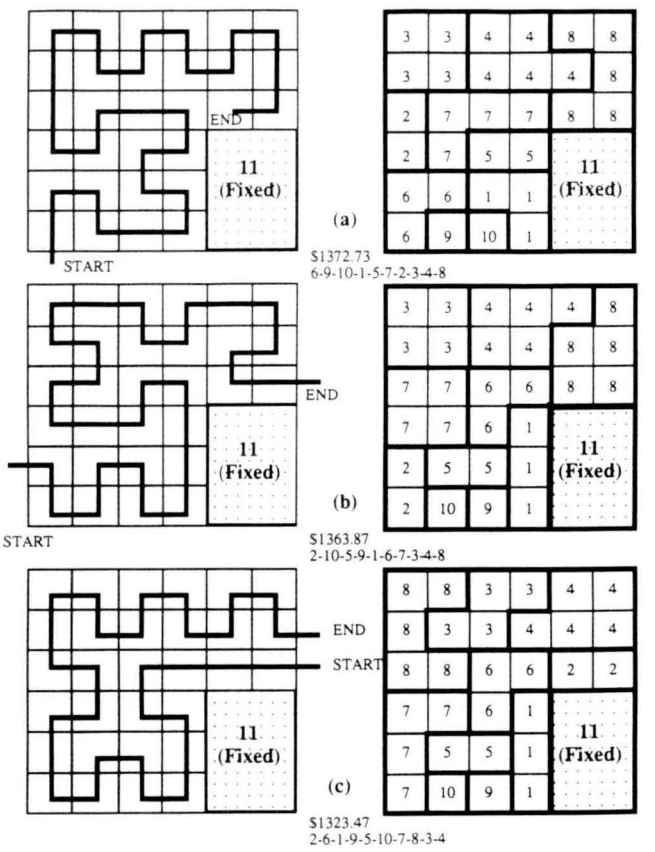


Table 4. There are four reasons behind the difference in runtimes. First, MULTIPLE considers more exchanges than CRAFT. Second, MULTIPLE evaluates the *actual cost* of each exchange, while CRAFT only estimates it. Third, although we did not impose any shape constraints in MULTIPLE (to keep the comparison equitable), the program automatically computes the shape measure for each department after each exchange. Finally, our copy of CRAFT was written in FORTRAN, while MULTIPLE is written in PASCAL. Benchmark problems we ran (on the above 386-PC) indicate that our FORTRAN compiler generates object codes that run about 2.25 times faster than its PASCAL counterpart. Furthermore, although the relative increase in runtime is substantial, the longest runtime (5.5 minutes) is well within reason to solve a 25-department problem on the 386-PC.

Although CRAFT has the ability to consider three-way exchanges, we ran it with only two-way exchanges to maintain equity. MULTIPLE can be modified to consider three-way exchanges. In fact, unlike CRAFT, MULTIPLE can exchange any three departments. Therefore, including three-way exchanges is likely to further improve the performance of MULTIPLE over CRAFT. However, as we discuss in the next section, there are more promising extensions to MULTIPLE.

7. Extensions to MULTIPLE

In MULTIPLE, the number of departments on each floor remains constant due to pairwise exchanges. With dummy departments (which have negligible area requirements and no flow), MULTIPLE can vary the number of (real) departments on a floor. We have found dummy departments to be effective and recommend placing several of them on each floor in the initial layout. One may also vary the number of departments on a floor by exchanging two departments on one floor for one department on another floor; i.e., a "2–for–1" exchange. In general, even with dummy departments, some 2–for–1 exchanges cannot be achieved through two or more two-way exchanges. Therefore, the cost of the final layout obtained with two-way and 2–for–1 exchanges is very likely to be less than that obtained with two-way exchanges only. Unfortunately, the number of all possible 2–for–1 exchanges increases rapidly with problem size.

Cost reductions that can be obtained with exchanges such as the 2–for–1 exchange, combined with MULTIPLE's inability to temporarily accept higher-cost solutions, suggests the use of a fundamentally different search procedure. We believe that extending MULTIPLE in a *simulated annealing* framework may be a very effective approach. Simulated annealing may reduce the path dependency of MULTIPLE. Also, since we use spacefilling curves, any (feasible) exchange can be accommodated simply as a new layout sequence on each floor. Thus, not only may 2–for–1 exchanges be considered along with 3–for–1 or 3–for–2, but combinations of n_1–for–n_2 exchanges may be performed in a single iteration. Of course, the type of exchanges to consider at each iteration and how to generate them is a topic for future research in multi-floor layout. Similar enhancements would apply to the single-floor layout problem as well.

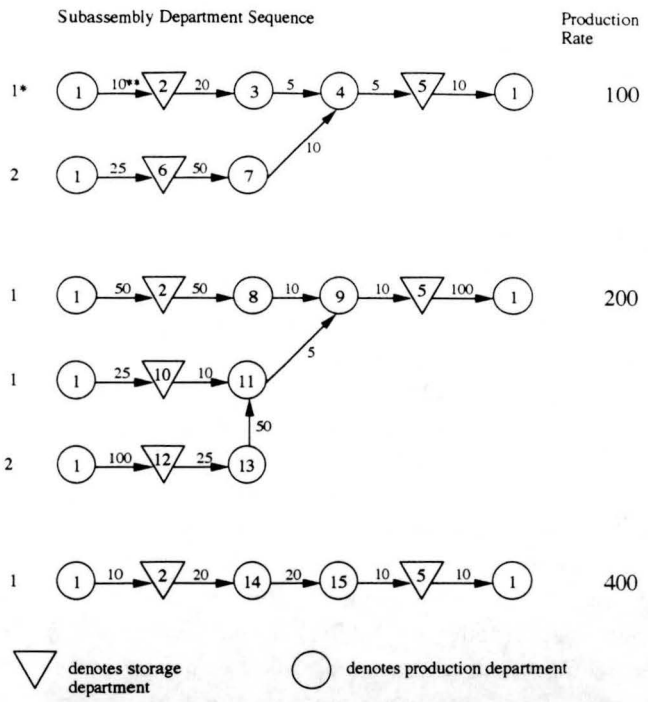
Table 4 **The Average, Standard Deviation, and Maximum Runtime (expressed in seconds) for CRAFT and MULTIPLE**

	Craft			Multiple		
Data Set	Average	Standard Deviation	Max	Average	Standard Deviation	Max
11	0.54	0.14	0.80	7.96	3.19	12.02
15	1.43	0.40	2.20	41.23	7.47	52.95
25	4.44	1.39	6.60	243.52	53.55	331.42

8. Implementation and Conclusions

A tailored version of MULTIPLE was implemented in a large, four-floor production facility with nearly 70 departments. The building was non-rectangular (with certain obstacles), and had several adjoining sections built at different time periods. Therefore, the floor loading capacity as well as the floor-to-ceiling distance varied from floor to floor, and from one section of the building to another. There were five elevators located at various points. Since management was reluctant to manually generate a 70 by 70 flow matrix, we developed a "flow matrix generator" which is a unique application of the "backward explosion" technique used in MRP systems. For each end product, we first determine the production

Figure 9 Sample Product Flow Chart to Generate a Flow Matrix



* Number in front of subassembly department sequence corresponds to the number of subassemblies in the final product.

** Number on top of the arrow between two departments corresponds to the size of the unit load moved between the two departments.

route and the number of subassemblies per end product. We next determine the basic units of flow. For example, in Figure 9 (where we have three end products), the subassembly of the first product flows from department 1 to 6 in 25 pieces / container. Hence, the material flow from department 1 to 6 is equal to $2(100/25) = 8$ containers / time unit. Repeating the same computation for each flow, we automatically generate the entire flow matrix. Ultimately, the flow matrix generator became a very useful tool. Simply by varying the production rates, the plant manager was able to generate alternative flows.

We also collected data on departmental area requirements. We were allowed to compress storage departments by up to 50%. Several shape and regional constraints were also considered. Given the nature of the building, we opted for hand-generated "spacefilling curves." Management also expressed their concern over the steepest-descent nature of MULTIPLE. Therefore, at each iteration, the program displayed the best twenty exchanges in decreasing order of layout cost savings. This enabled the analyst to evaluate each exchange in terms of relocation versus layout costs. Entering realistic relocation costs *a priori* (as in Hicks and Cowen 1976) was not considered to be practical.

The tailored version of MULTIPLE was implemented on a 20-MHz DOS-based 386PC (located at the plant site) with a 387 coprocessor and a DOS-Extender. (The latter was required due to the large problem size.) Each iteration of the tailored algorithm required about 25 minutes. Given the portability and ease-of-use offered by PCs, the above runtime was considered reasonable. Using the tailored version of MULTIPLE and working jointly with the plant technical staff, we identified several cost-justified layout improvements.

In conclusion, MULTIPLE offers several advantages over CRAFT in single-floor applications. It also generates reasonably good layouts for multiple floor facilities without splitting departments. MULTIPLE's primary strength is derived from its use of spacefilling curves to rapidly relayout one or two floors after an exchange. These curves also make it possible to consider more powerful exchange routines than two-way or three-way exchanges. Two other key features of MULTIPLE is its ability to effectively handle a range of area requirements for each department and its use of a "new" shape measure to avoid irregular department shapes. Although MULTIPLE is based on the "conventional" assumption of travel between department centroids, it can be easily modified to accommodate alternative measures of layout efficiency. Lastly, MULTIPLE explicitly considers the location of each existing and potential lift; however, it does not consider the throughput capacity of the lifts. This aspect of the problem is subject to further research.[1,2]

[1] For research purposes only and within reasonable boundaries, the authors are willing to run the current version of MULTIPLE with data submitted on IBM-PC or compatible disks. Interested readers may contact the authors for data format and further information.

[2] The authors would like to thank Professor Roger V. Johnson for providing an updated copy of SPACECRAFT which was used in §5. This study was partially supported by Dr. Bozer's Presidential Young Investigator Award under NSF Grant DDM-8858562 and a research grant from General Electric Company, DRDA 90-0083.

Figure A1 Procedure to Generate a Spacefilling Curve for an r by c Rectangle

```
Procedure SFC (r × c)
  find max k such that (2^k ≤ c)
  find max m such that (2^(m-1) ≤ r)
  n = min (k, m)
  if n = 0, then
        only one possible solution, Return
  else
        Continue
  draw 2^(n-1) × 2^(n-1) Hilbert Curve beginning at point 1 (Figure A2)
        if r − 2^(n-1) > 0, then
            end Hilbert Curve at point 2 (Figure A2)
            SFC ( min {r − 2^(n-1), 2^(n-1) } ; max { r − 2^(n-1), 2^(n-1) })
        else
            end Hilbert Curve at point 3 (Figure A2)
  mirror image r × 2^(n-1) A onto r × 2^(n-1) B
  if c > 2^n, then
        SFC ( min {r , c − 2^n } ; max {r, c − 2^n })
  else
        Return
End Procedure SFC
```

```
c must be ≥ r; if not reorient;
defining areas, A, B & C
in Figure A2;

degenerate SFC (1 × 1) − point;

decide where curve will end
based on r and c values;

recursively call SFC with additional
area above curve;

now A and B filled;
recursive call to SFC for C;

if C is empty, then
return to last SFC call;
```

Figure A2 The Rectangles A, B, and C as Defined by the SFC Procedure

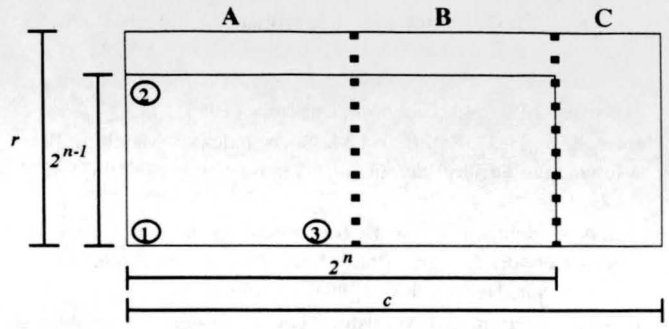
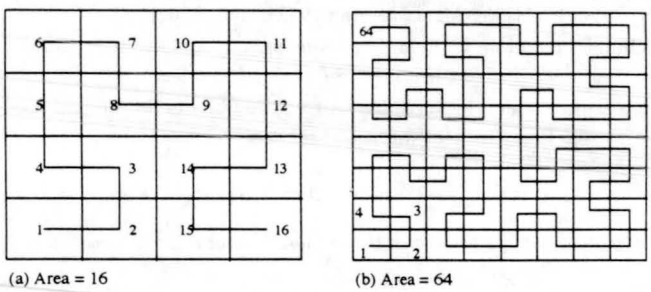


Figure A3 Examples of the Hilbert Curve (1891)

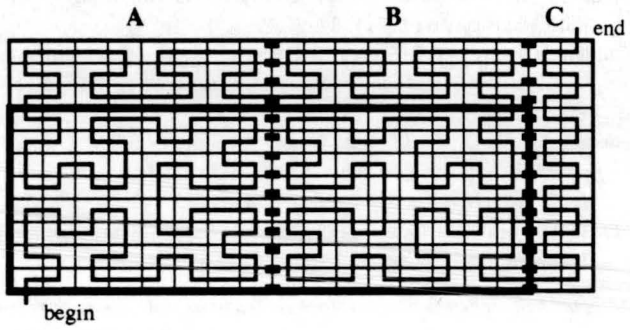


(a) Area = 16 (b) Area = 64

Appendix

A spacefilling curve for a rectangle may be generated by the procedure shown in Figure A1. The rectangle is defined by r rows and c columns, while n (restricted to integer values) is determined such that the 2^{n-1} by 2^n rectangle (shown in Figure A2) is the largest rectangle to be contained in the r by c area. (If c is not greater than or equal to r, the rectangle should be reoriented.) Once n is determined, rectangles A, B, and C are defined. If $r = c = 1$, both rectangles B and C are undefined and the procedure simply connects the appropriate points.

Figure A4 Spacefilling Curve Generated for an 11 by 18 Rectangle



- Originally, $n = 4$

The original call to the SFC procedure will fill rectangle A with the appropriate Hilbert Curve (see Hobson 1950 and Figure A3). Once rectangle A is filled, it will mirror image itself onto rectangle B, and then call the SFC procedure recursively to fill rectangle C. The difference between r and 2^{n-1} determines whether the spacefilling curve that begins at point 1, ends at point 2 or point 3. If the curve ends at point 2, then the SFC procedure will be called recursively to fill the remainder of rectangle A. Finally, each successive recursive call to the SFC procedure will link point 2 (or 3) from the previous rectangle to point 1 in the recursively defined rectangle. Figure A4 shows the spacefilling curve generated when the SFC procedure is applied to an 11 by 18 rectangle. It also shows rectangles A, B, and C obtained with $n = 4$.

References

Armour, G. C. and E. S. Buffa, "A Heuristic Algorithm and Simulation Approach to Relative Location of Facilities," *Management Sci.*, 9, 2 (1963), 294–309.

Bartholdi, J. J. and L. K. Platzman, "An $O(n \log n)$ Planar Traveling Salesman Heuristic Based on Spacefilling Curves," *Operations Res. Letters*, 1, 4 (1982), 121–125.

—— and ——, "Design of Efficient Bin-Numbering Schemes for Warehouses," *Material Flow*, 4 (1988), 247–254.

Buffa, E. S., G. C. Armour and T. E. Vollman, "Allocating Facilities with CRAFT," *Harvard Business Review*, 42 (1964), 136–158.

Chisman, J. A., "The Clustered Traveling Salesman Problem," *Computers and Operations Res.*, 2 (1975), 115–119.

Çınar, Ü., "Facilities Planning: A Systems Analysis and Space Allocation Approach," *Spatial Synthesis in Computer-Aided Building Design*, Charles M. Eastman (Ed.), Wiley, 1975, 19–40.

Donaghey, C. E. and V. F. Pire, "Solving the Facility Layout Problem with BLOCPLAN," Industrial Engineering Department, University of Houston, TX, 1990.

Drezner, Z., "DISCON: A New Method for the Layout Problem," *Operations Res.*, 20 (1980), 1375–1384.

Foulds, L. R., "Techniques for Facilities Layout: Deciding Which Pairs of Activities Should be Adjacent," *Management Sci.*, 29 (1983), 1414–1426.

—— and D. F. Robinson, "Graph Theoretic Heuristics for the Plant Layout Problem," *International J. of Production Res.*, 16 (1978), 27–37.

Freeman, H., "Computer Processing of Line-Drawing Images," *Computing Surveys*, 6 (1974), 57–97.

Golden, B. L. and W. R. Stewart, "Empirical Analysis of Heuristics," *The Traveling Salesman Problem*, Lawler, Lenstra, Rinnooy Kan, Shmoys (Eds.), Wiley (1985), 207–214.

Hassan, M. M. D., G. L. Hogg and D. R. Smith, "SHAPE: A Construction Algorithm for Area Placement Evaluation," *International J. of Production Res.*, 24 (1986), 1283–1295.

Hicks, P. E. and T. E. Cowen, "CRAFT-M for Layout Rearrangement," *Industrial Engineering*, 8 (1976), 30–35.

Hobson, E. W., *The Theory of Functions of a Real Variable and the Theory of Fourier's Series, Volume I*, 3rd Edition, Cambridge University Press and Harren Press, Washington D.C., 1950.

Industrial Engineering, "Candy Maker 'Sweetens' Efficiency with Pflow Lifts," 22, 10 (1990a), 60.

——, "Vertical Lift Grows with New Distribution Center," 22, 9 (1990b), 81.

Jacobs, F. R., "A Note on SPACECRAFT for Multi-Floor Layout Planning," *Management Sci.*, 30, 5 (1984), 648–649.

Johnson, R. V., "SPACECRAFT for Multi-Floor Layout Planning," *Management Sci.*, 28, 4 (1982), 407–417.

Kaku, K., G. L. Thompson, and I. Baybars, "A Heuristic Method for the Multi-Story Layout Problem," *European J. of Operational Res.*, 37 (1988), 384–397.

King, J. and R. E. Johnson, "Silk Purses From Old Plants," *Harvard Business Review*, 61, 2 (1983), 147–156.

Lew, P. and P. H. Brown, "Evaluation and Modification of CRAFT for an Architectural Methodology," *Proceedings of the Design Methods Group First International Conference*, Gary T. Moore (Ed.) (1968), 155–161.

Liggett, R. S. and W. J. Mitchell, "Optimal Space Planning in Practice," *Computer Aided Design*, 13 (1981), 277–288.

Material Handling Engineering, "Plant Renovation: The Low-Cost Road to Success," 43, 2 (1988a), 41–51.

——, "Vertical Conveyor Eliminates Need for New Construction," 40, 7 (1988b), 84–86.

——, "Vertical Reciprocating Conveyors: Flexible Handling Devices," 45, 9 (1990), 81–85.

Meller, R. D., "Layout Algorithms for Single and Multiple Floor Facilities," Ph.D. Dissertation, Department of Industrial and Operations Engineering, The University of Michigan 1992.

Modern Materials Handling, "Vertical Conveyors Increase Plant's Efficiency 33%," 40, Casebook Directory (1985), 113.

Montreuil, B., H. D. Ratliff and M. Goetschalckx, "Matching Based Interactive Facility Layout," *IIE Transactions*, 19 (1987), 271–279.

——, "A Modeling Framework for Integrating Layout Design and Flow Network Design," *Proceedings of the Material Handling Res. Colloquium*, Hebron, KY (1990), 43–58.

Pire, V. F., "Automated Multistory Layout System," Unpublished Master's thesis, Industrial Engineering Department, University of Houston, Houston, TX, 1987.

Ritzman, L. P., "The Efficiency of Computer Algorithms for Plant Layout," *Management Sci.*, 18 (1972), 240–248.

Seehof, J. M. and W. O. Evans, "Automated Layout Design Program," *J. of Industrial Engineering*, 18 (1967), 690–695.

Tompkins, J. A. and R. Reed, Jr., "An Applied Model for the Facilities Design Problem," *International J. of Production Res.*, 14, 5 (1976), 583–595.

—— and J. A. White, *Facilities Planning*, Wiley, New York, 1984.

Accepted by Stephen C. Graves, former Departmental Editor; received May 1991. *This paper has been with the authors* 6 *months for* 2 *revisions.*