A CONSTRUCTION-TYPE LAYOUT ALGORITHM FOR MULTI-FLOOR FACILITIES WITH CAPACITATED LIFTS

Yavuz A. Bozer
Department of Industrial & Operations Engineering
University of Michigan
Ann Arbor, MI 48109-2117

Russell D. Meller
Department of Industrial Engineering
Auburn University
Auburn, AL 36849-5346

Technical Report 92-47

August 1992

A Construction–Type Layout Algorithm for Multi–Floor Facilities with Capacitated Lifts†

Yavuz A. Bozer
Department of Industrial and Operations Engineering
The University of Michigan

Russell D. Meller
Department of Industrial Engineering
Auburn University

August 26, 1992

Abstract

In this paper we present a construction–type layout algorithm for multi–floor manufacturing facilities. We also define and heuristically solve the lift location–allocation problem which is concerned with the lift system configuration, i.e., the number, location, and workload of the vertical handling devices in the facility. The overall approach is composed of three stages. In the first stage, we use a mixed–integer formulation to optimally assign departments to floors. This linear formulation exploits the structure of the vertical distance function, and the objective is to minimize the amount of vertical handling in the facility. In the second stage, we use simulated annealing to develop the layout of each floor concurrently. In doing so, we assume that all existing (or potential) lifts designated by the user are "open." The above two stages improve upon existing multi–floor facility layout algorithms and produce low–cost solutions to data sets presented previously in the literature. In the third stage, we consider the the throughput capacity of each lift, and again use simulated annealing to determine the lift system configuration for the layout obtained in the second stage. The objective is to minimize the third–stage cost, which is composed of amortized lift costs and expected waiting time costs at (open) lifts. Lastly, to evaluate the above three–stage approach, we solve several test problems to empirically show that it produces optimal or near–optimal solutions with respect to the global–optimal, which is obtained by simultaneously solving the layout and the lift location–allocation problems.

---

†This material is based on work supported by the National Science Foundation under Grant No. DDM–8858562.

1 Introduction

The *facility layout problem* is defined as arranging departments within a facility to minimize, subject to constraints, the interaction cost between the departments. (The interaction cost between two departments is expressed as the flow times the distance between the departments.) In general, the departments have unequal area requirements, and some of them may be constrained to certain locations within the facility *a priori.*

The quadratic assignment problem (QAP) formulation of the (single-floor) facility layout problem is NP–complete [6]. As a result, many heuristics have been developed for the single-floor facility layout problem. The *multi-floor facility layout problem*, on the other hand, encompasses all aspects of the single-floor facility layout problem, and in addition, it includes vertical flows and area constraints for individual floors. (Vertical flows are required to use a *lift*, which we define here as a generic vertical material handling device.) Moreover, since we assume that departments cannot be split across floors, some layouts may not be *area feasible* in a multi-floor problem.

Before formalizing the above differences mathematically, consider the following problem parameters (whose values are supplied by the analyst):

$$\begin{aligned} f_{ij} &= \text{the flow from department } i \text{ to department } j; \\ c_{ij}^H(c_{ij}^V) &= \text{the horizontal (vertical) unit cost to move one unit of flow from } i \text{ to } j; \\ a_i &= \text{the minimum required area for department } i; \\ A_k &= \text{the maximum usable floor space available on floor } k. \end{aligned}$$

The horizontal and vertical distance between departments i and j, i.e., d_{ij}^H and d_{ij}^V, respectively, are obtained from the layout. Thus, the objective function of the multi-floor facility layout problem is given by:

$$\min \sum_i \sum_j (c_{ij}^H d_{ij}^H + c_{ij}^V d_{ij}^V) f_{ij}. \tag{1}$$

In addition to those constraints used in its single-floor counterpart, the following constraints must be used with the multi-floor layout problem:

$$\sum_k x_{ik} = 1 \quad \forall i \tag{2}$$

$$\sum_i a_i x_{ik} \leq A_k \quad \forall k \tag{3}$$

where x_{ik} equals one if department i is assigned to floor k, and zero otherwise. (The closed-form expression of the constraints used in a single-floor layout problem depends on the type of formulation adopted; see [14] and [18], among others.)

With multi-floor facilities, in addition to the arrangement of departments on each floor, one must address the *lift location-allocation problem* (LLAP). The LLAP is defined as determining which existing (or potential) lifts in a facility are to be "opened," and the

assignment of each vertical flow to one of the open lifts. This problem will be discussed in more detail in §6.

The contribution of this paper is the development of a three-stage algorithm to solve the multi-floor facility layout problem as defined above. In addition to the first two stages (where we develop a new construction-type layout algorithm), in the third stage we explicitly address the LLAP, which makes our approach more comprehensive and realistic than prior solution procedures. The primary application area of the algorithm presented in this paper is manufacturing (or production/distribution) facilities. As such, we envision that the problem will involve no more than, say, 40 or 50 departments arranged in a building with no more than, say, 4 or 5 floors, and 5 or 6 lifts. Here we are not concerned with determining the layout of a "tall" office building with potentially hundreds of departments and a large number of floors. (For reasons described in §2, much of the previous research results in multi-floor facility layout are more applicable to office buildings than to manufacturing facilities.) We are also not concerned with people flow or one-time equipment transfers between the floors; i.e., the lifts in question are dedicated to production-related daily material flow between the floors. We assume that the flow is constant; Rosenblatt and Kropp [20] show that the single-period stochastic layout problem may be solved with algorithms that are based on such an assumption if the objective is to minimize the *expected* layout cost.

Using the construction-type layout algorithm we develop in this paper, we were also able to numerically show that, in some cases, one may indeed achieve a lower *layout cost* with a multi-floor facility than with a single-floor facility. To our knowledge, such a comparison between single and multi-floor manufacturing facilities is the first of its kind.

2 Literature Review

A limited number of heuristic procedures have been developed to solve the multi-floor facility layout problem. Existing algorithms can be divided into two types: improvement and construction. Improvement-type algorithms begin with an initial layout and attempt to improve it by exchanging department locations. In contrast, construction-type algorithms attempt to construct a layout with no reference to an initial solution. In general, improvement procedures are used when a current layout exists, and construction procedures are used when there is a "green field" application, a vacant facility, or a layout is to be determined without necessarily imposing an initial layout constraint. Of course, one may always construct "composite heuristics" by applying an improvement procedure to the solution obtained from a construction procedure.

Consider first improvement procedures. SPACECRAFT [12] by Johnson, extends the (single-floor) improvement-type algorithm CRAFT [1] to multi-floor problems. SPACECRAFT is limited to exchanging only equal-area departments across floors unless departments can be split across two or more floors. MULTIPLE [3] by Bozer, Meller and

Erlebacher, removes this limitation by the use of spacefilling curves. SABLE [17] by Meller and Bozer, generalizes MULTIPLE's exchange routine and incorporates a simulated-annealing based search strategy.

In SABLE, as in MULTIPLE, a layout is represented as a sequence of departments and a spacefilling curve. Departments are placed in the facility following the sequence of departments, and their location in the facility is determined by a user-specified spacefilling curve for each floor. SABLE generates candidate department sequences by allowing each department to potentially change its position in the current sequence. Such a procedure generates any number or type of department exchanges; we will demonstrate it in §5.2. In addition, by accepting or rejecting candidate sequences in a simulated-annealing based search, SABLE achieves lower-cost solutions than solutions obtained previously for single and multi-floor test problems [17]. Since the study of improvement procedures is not our main focus here, the reader may refer to the above papers for further details.

Consider next multi-floor construction algorithms, which include ALDEP [21] by Seehof and Evans, BLOCPLAN [5] by Donaghey and Pire, SPS [15] by Liggett and Mitchell, and MSLP [13] by Kaku, Thompson and Baybars. The above algorithms employ a two-stage approach: in the first stage each department is heuristically assigned to a particular floor, and in the second stage the layout of each floor is developed independently.

ALDEP and BLOCPLAN both ignore vertical flow once departments have been assigned to floors. It is unclear how ALDEP makes this assignment, but departments may be split across two or three floors to maintain area feasibility on each floor. BLOCPLAN uses heuristic methods (based on adjacency) to assign departments to floors (see Pire [19]). Only those departments that are located on the same floor are considered adjacent. The heuristic used in BLOCPLAN, like other adjacency-based heuristics, does not reflect the degree to which departments are not adjacent; i.e., the horizontal distance and/or the number of floors that separate non-adjacent departments is not measured. Also, neither ALDEP nor BLOCPLAN considers existing or potential lift locations in the facility.

In SPS it is assumed that there is only one lift (or a centrally located single group of lifts). Such an assumption may be appropriate for office buildings, but in manufacturing facilities (material handling) lifts are typically provided at several locations. SPS implements the two-stage approach (defined above) by formulating each stage as a QAP, which is solved by applying an expected value method proposed by Graves and Whinston [7]. In the first stage a department may be split between two floors due to this particular method. In the second stage, the layout of each floor is determined independently of the other floors. However, since there is only one lift, departments that have high levels of interaction with departments on other floors are likely to be placed close to the lift.

MSLP is similar to SPS: it is based on a two-stage approach, it assumes one lift location, and in the second stage it constructs the layout of each floor independently. However, in MSLP: (1) it is assumed that all the departments are equal in area, (2) the assignment of departments to K floors is solved as a K-median problem, and (3) an improvement heuristic is applied after the floor layouts are determined by solving a QAP

for each floor. (The improvement heuristic considers exchanging only those departments that are located on different floors.) As acknowledged by the authors [13], extending MSLP to the general case where departments have unequal area requirements is not straightforward.

In short, although a few multi-floor construction-type layout algorithms have been developed, there are certain factors which would limit their use in a multi-floor manufacturing facility. Vertical flow and lift locations are not considered in some of the algorithms and when they are, multiple lift locations are not allowed. When assigning (unequal-area) departments to floors, some departments may be split across two or more floors. In facility layout, a department is the smallest non-divisible entity, by definition. Also, splitting a department creates additional vertical flow – between the pieces – which is not accounted for in the objective function. Furthermore, due to the machinery involved, or other reasons such as supervision, quality control, timely feedback, etc., splitting a department may be infeasible or highly undesirable. Finally, all the existing algorithms determine the layout of each floor independently; this is likely to lead to inferior solutions especially if there are multiple lift locations.

Assuming equal distances between adjacent floors, the three-stage algorithm we develop in this paper addresses the above shortcomings in the first two stages. In the third stage, we explicitly consider lift costs and throughput capacities, which are not considered in the above algorithms. Unlike previous studies, we also compare our results to globally-optimal solutions obtained by solving the three stages concurrently.

3 Outline of Proposed Algorithm

As in existing algorithms, in the first stage of the proposed algorithm we assign each department to a floor, and in the second stage we determine the layout of each floor. However, the algorithm we propose is superior to existing algorithms since we present a new *linear* mixed-integer programming formulation to optimally solve the first-stage problem, and we develop the second-stage floor layouts concurrently rather than independently. Furthermore, we do not make any restrictive assumptions about department areas or the number and locations of lifts. Departmental area requirements are specified by the user, who also designates all existing or potential lift locations in the facility.

The first stage is motivated by the observation that in many application areas, including manufacturing, it is generally undesirable to locate two departments with a high level of interaction on different floors. Reasons for this include quantifiable costs, such as vertical material handling and capital investment in lifts, as well as costs that are not easy to quantify, including line-of-sight management and timely feedback. Thus, determining the optimal department-to-floor assignments in the first stage is likely to lead to good solutions to the overall problem. The objective in the first stage is to minimize the overall vertical handling requirement in the facility regardless of which lifts are used.

Assuming that all the existing/potential lift locations designated by the user are "open," in the second stage we determine the layout of each floor concurrently. As described later, the flow between departments on different floors is assumed to be handled through the lift which minimizes the total horizontal distance traveled. (The expected waiting times at the lifts are not taken into account at this stage.)

In the third stage, we take the (fixed) layout from stage two, and given the amortized cost of the lifts as well as the cost associated with waiting at each lift, we solve the LLAP which seeks a minimum-cost solution. (In determining the expected waiting times at each lift we use an analytical model developed by Cho [4].) The LLAP is an important problem that, ideally, should be solved in conjunction with the layout problem. However, the complexity of the two problems does not allow such an approach. We propose to solve the LLAP *after* the layout of the facility has been determined since an efficient layout reduces the material handling effort in the facility. Also, without a layout it is difficult to determine the workload on individual lifts.

A new formulation to obtain the optimal solution to the first stage is shown in §4, and the heuristic algorithm for the second stage is shown in §5. The solution procedure for the LLAP is presented in §6. The performance of the proposed approach on layout and LLAP test problems is presented in §7. Also in §7, we discuss the performance of the three-stage approach relative to the globally-optimal solution, which simultaneously solves the layout and LLAP problems. We present our conclusions in §8, which includes a comparison of *layout costs* between a single and a multiple floor facility.

4 Assigning Departments to Floors

The objective in this first stage is to determine the optimal department-to-floor assignments (with no splitting) in order to minimize the overall vertical handling requirement in the facility regardless of lift usage. Prior studies (mainly [13] and [15]) use a quadratic objective function, which necessitates heuristic procedures for problems of reasonable size. The objective function of the floor assignment formulation we present here is linear because we exploit the structure of the vertical distance function, which is simply a multiple of an assumed constant distance between adjacent floors.

4.1 Definitions and Notation

The following notation will be used in the floor assignment formulation.

- The indices i, j will be used for departments, where $i, j = 1, \ldots, N$.
- The indices g, k will be used for floors, where $g, k = 1, \ldots, K$.

- The vertical distance between departments i and j when they are assigned to floors k and g, respectively, is denoted by d_{ik}^{jg}.

In addition to the parameters described in §1, the following parameters are given:

1. The distance between any two adjacent floors is δ units.
2. The set of non-zero flows[1] is denoted by $F = \{f_{ij}\}$. That is, $|f_{ij}| > 0 \forall i, j \in F$ and furthermore, $|F| = M$.
3. The mth non-zero flow, f_m, originates from department $i(m)$ and terminates at department $j(m)$.

4.2 A Linear Formulation

Previous QAP procedures used for solving the first-stage problem make no assumptions about the structure of the distance function. With the distance between adjacent floors equal, however, the vertical distance function has the following form:

$$d_{ik}^{jg} = \delta|k - g|. \tag{4}$$

Recall that x_{ik} equals one if department i is assigned to floor k, and zero otherwise. Thus, if we use the following constraint to assign each department a floor number,

$$y_i = \sum_k k x_{ik}, \tag{5}$$

the vertical distance function becomes:

$$d_{ik}^{jg} = \delta|y_i - y_j|. \tag{6}$$

The above observation forms the foundation of the linear formulation. Although there are other linearization schemes developed for the QAP, such schemes are "generic" ones which typically increase the number of binary variables. In our approach, the x_{ik}'s are still the only binary variables we need because each y_i implicitly assumes an integer value for binary x_{ik}'s.

The total vertical handling requirements is represented as the sum of the *revised flows.* (The revised flow from department $i(m)$ to department $j(m)$, say, f'_m, is the product of the flow from department $i(m)$ to department $j(m)$ and the distance between the floors to which departments $i(m)$ and $j(m)$ have been assigned.) In other words, in the following (linear) model we simply minimize the sum of the vertical flows times vertical distances while observing floor and department area constraints:

[1]Negative flow values are permitted to enable the model to separate particular departments.

$$\min \quad \sum_{m=1}^{M} f'_m \tag{7}$$

$$\text{s.t.} \quad \sum_{k=1}^{K} k x_{ik} = y_i \qquad i = 1, \ldots, N \tag{8}$$

$$f'_m \geq (y_{i(m)} - y_{j(m)}) \delta f_m \qquad m = 1, \ldots, M \tag{9}$$

$$f'_m \geq (y_{j(m)} - y_{i(m)}) \delta f_m \qquad m = 1, \ldots, M \tag{10}$$

$$\sum_{k=1}^{K} x_{ik} = 1 \qquad i = 1, \ldots, N \tag{11}$$

$$\sum_{i=1}^{N} a_i x_{ik} \leq A_k \qquad k = 1, \ldots, K \tag{12}$$

$$\text{where} \quad x_{ik} = \begin{cases} 1 & \text{if department } i \text{ is assigned to floor } k, \\ 0 & \text{otherwise} \end{cases} \tag{13}$$

The above formulation defined by expressions (7) - (13) will be referred to as the floor assignment formulation (or FAF). Note that constraint sets (9) and (10) are used to determine the vertical distance between two departments and the revised flows utilized in the objective function. The number of binary variables in FAF is equal to NK, and there are N multiple-choice constraints (given by constraint set (11)). There are $N + M$ continuous variables.

A branch-and-bound algorithm — developed from the subroutines of the Optimization Subroutine Library (OSL) [9] — was used to solve FAF. The performance of this algorithm was enhanced considerably by explicitly considering the multiple-choice constraints. FAF determines the department-to-floor assignments and the resulting vertical flow in the facility. However, the total layout cost also depends on horizontal, as well as the vertical, distances between the departments. The former are determined by the layout of each floor. The procedure we developed for determining the floor layouts is discussed in the next section.

5 Determining the Floor Layouts

The procedure presented in this section solves the second-stage problem, which is concerned with determining the layout of each floor (for *fixed* department-to-floor assignments obtained in the first stage). We assume that all existing/potential lift locations are specified by the user *a priori* and that all the corresponding lifts are "open." For the purposes of comparing our procedure with other layout algorithms, we also assume throughout the paper that the horizontal distance between two departments is measured rectilinearly between department centroids. Likewise, since lift capacities are not considered in other

approaches (and will be considered in the third stage of our approach), we assume that the vertical flow between two departments is handled via the lift which minimizes the horizontal travel distance between the two departments; that is,

$$d_{ij}^H = \min_{\ell}(d_{i\ell}^H + d_{\ell j}^H), \tag{14}$$

where the index ℓ designates a lift, and $d_{i\ell}^H$ designates the horizontal distance from the centroid of department i to lift ℓ. Determining the layout of each floor concurrently is especially important when there are multiple lift locations. As discussed below, with a single lift, floor layouts can be constructed independently.

5.1 A Construction Approach

Let us momentarily assume that there is one lift in the facility and that its location is known. Then for each floor k, a new flow matrix, F^k, may be constructed. This flow matrix consists of the flows between departments assigned to floor k and an additional column and row which corresponds to the flows between each department on floor k and the lift. (Essentially, a "department" that represents the lift is created on each floor and the flows between departments located on different floors are assigned to the "lift department.") Recall that y_i denotes the floor number of department i and f_{ij} is an element of the flow matrix F. Hence, F^k is defined mathematically as:

$$F^k = [f_{ij}^k] = \begin{cases} f_{ij} & \text{if } y_i = y_j = k, \quad \text{(a)} \\ f_{i\ell} & \text{if } y_i = k, y_j \neq k, \quad \text{(b)} \\ f_{\ell j} & \text{if } y_i \neq k, y_j = k. \quad \text{(c)} \end{cases} \tag{15}$$

Each floor layout may now be determined independently, without loss of solution quality.

A traditional single-floor layout construction heuristic would add departments, one at a time, to the partial layout based on their flow values and the location of previously placed departments. With one lift location, the lift is the first "department" to be placed in the layout, and as shown above, the flow between the lift and the entering departments is defined by (15b) and (15c). However, if there is more than one lift location (and nearly all production facilities have multiple lift locations), the flow between a department and a particular lift depends on both the origin and the destination of the flow. Since all the department locations are not known, whether or not a particular department has an interaction with a particular lift cannot be determined *a priori*. Therefore, if a traditional construction-type approach were implemented to solve the second stage single-floor layout problems, the horizontal distances between departments (due to travel to/from the lifts) may increase unnecessarily.

In contrast, if a starting point is provided, i.e., an "arbitrary" layout for each floor, then one may determine the appropriate lift to be used by each vertical flow in the facility. Also, if this starting point is revised, i.e., the locations of two or more departments on one

or several floors are exchanged, one may still determine the appropriate lifts to be used in the revised layout. Hence, with multiple lift locations, one may use an improvement-like procedure to determine the floor layouts.

5.2 An Improvement-Type Algorithm

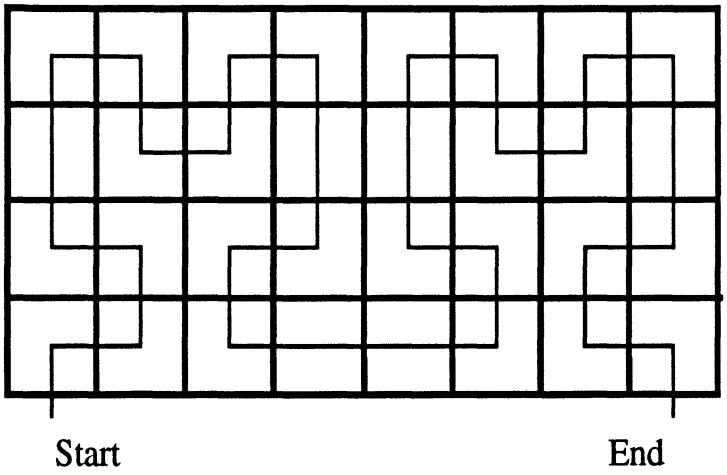
Using an improvement-type algorithm in the second stage, we determine all floor layouts concurrently, while considering multiple lift locations and observing the department-to-floor assignments made in the first stage. Given an arbitrary initial layout for each floor, we generate revised layouts that change the locations of some non-fixed departments within one or more floors. Since we do not exchange departments across floors, the improvement-type algorithm we propose maintains the minimum vertical handling cost determined in the first stage and attempts to minimize the horizontal handling cost by revising the floor layouts (refer to objective function given earlier by (1)).

We specify a layout as a *sequence of department numbers* with *dividers* to designate the floors. For example, in a 12-department, 3-floor facility, the sequence 4 - 1 - 7 - 12 - 10 / 8 - 6 - 11 / 5 - 3 - 9 - 2 indicates that departments 4, 1, 7, 12, and 10 have been assigned to the first floor, departments 8, 6, and 11 to the second floor, and so on. Given such a sequence (i.e., a *layout vector*), the layout of each floor is constructed by using the procedure presented in [3] and [17]. This procedure assumes that a spacefilling curve has been defined for each floor. (See [3] for a discussion on spacefilling curve generation.) Given a curve for each floor, a layout vector, and the area data for the departments, a multi-floor layout can be rapidly constructed [3].

For example, consider the above layout vector. Given the spacefilling curve shown in Figure 1(a), for the first floor one would obtain the layout shown in Figure 1(b). The number of grids assigned to each department in Figure 1(b) is determined by the departmental area requirements, which is specified by the user. The layout of the other floors would be determined in a similar manner (using the same or a different curve).

Consider next the *address vector*, $\boldsymbol{a}$, which determines the position of each department in the layout vector. Each department i has a corresponding address in $\boldsymbol{a}$, $a(i)$, which is the sum of a fixed component and a variable component. The fixed component of $a(i)$ equals y_i, the floor to which department i is assigned. The variable component equals a value in the interval (0,1). Generating variable-component values and sorting the departments with respect to their $a(i)$ values yields a layout vector. (Note that the divider locations will remain unchanged due to the fixed components in $\boldsymbol{a}$.)

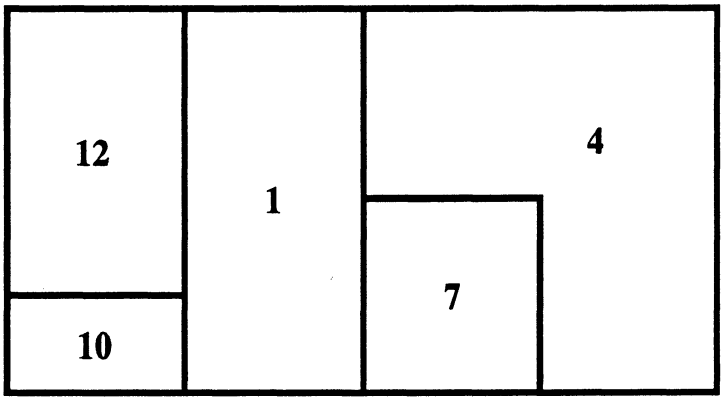
Revised (i.e., "candidate") layouts are generated (and accepted or rejected) according to a simulated-annealing based search procedure. (The theoretical properties of such a search strategy are explored in [11], and implemented with success in [8], [10], [17], [22], among others.) The manner in which we generate the layout vector is very similar to that shown in SABLE [17]. Each department i has a variable $b(i)$ associated with



(a)

4	4	4	4	7	7	12	12
4	4	4	4	7	7	12	12
4	4	1	1	1	1	12	12
4	4	1	1	1	1	10	10

(b)



(c)

Figure 1: Constructing Layouts with Various Layout Vectors.

it, where $b(i)$ is distributed $U(0,1)$. First, we randomly sample the $b(i)$ values. For each i, if the value of $b(i)$ is greater than some *critical value* specified by the user, then we randomly generate a new variable-component value for $a(i)$. Otherwise, the current $a(i)$ value remains unchanged. By sorting the layout vector with respect to the (new) department addresses, we obtain a candidate layout vector. For example, after applying the above procedure to the layout vector shown earlier, one may obtain the following candidate layout vector: 10 - 12 - 1 - 7 - 4 / 6 - 8 - 11 / 3 - 5 - 9 - 2, and the first-floor layout shown in Figure 1(c) without the underlying grid representation. Note that the user-specified critical value controls how different the layout vector is from one trial to the next [17].

The determination of an initial temperature (t_0), and the approach we use to control the overall annealing schedule (including temperature reduction and establishing equilibrium), are the same as the approach used in [17]. We next present the necessary notation and the simulated annealing algorithm.

5.3 Notation

The following notation will be used to present the Simulated-Annealing Based Algorithm for the Second Stage, that is, SABASS. When appropriate, variables were used in a manner similar to their use in [17]. Let

s^0 = initial layout vector — when combined with the floor assignment variables, spacefilling curve, and area data, it uniquely represents the initial layout.

s^* = "current best" layout vector which corresponds to the lowest cost layout identified by the algorithm.

s = the current layout vector.

s' = the candidate layout vector.

T = $\{t_1, t_2, t_3, \ldots\}$ — a set of annealing schedule temperatures, where $t_i = t_0(0.8)^{i-1}, \forall i > 1$.

t_0 = initial temperature.

e = epoch length — fixed number of candidate layout vectors that will be accepted during each epoch.

$f_j(s)$ = objective value of the jth accepted candidate layout vector, s.

$\overline{f}_e$ = $\left(\sum_{j=1}^{e} f_j(s)\right)/e$ — the mean objective function value of an epoch with e accepted candidate layout vectors.

$\overline{f}'_e$ = the overall mean of objective function values for the layout vectors accepted during the epochs previous to the current one for a given temperature.

ϵ_i = an error constant used to determine whether the system is in equilibrium at temperature i.

$\boldsymbol{a}$ = a vector of random variables that designates the address for each department in the layout vector, where each element equals the floor number of the corresponding department plus a $U(0,1)$ random variable.

$\boldsymbol{b}$ = a vector of $U(0,1)$ random variables that determines if the department addresses change.

β = critical value to determine whether a new address is assigned to a department, $\beta \in (0,1)$.

M = a number that is specified *a priori* to control the maximum number of epochs considered.

I = a counter to record the temperature setting which produced the "current best" layout vector, s^*.

N = the maximum number of successive temperature settings that do not produce a new s^*, $I \leq N$.

5.4 The Algorithm (SABASS)

The variables $t_1, e, \epsilon, \beta, M$ and N are control variables that are input by the user. The settings used for the control variables will be described in subsequent sections. A general outline of SABASS follows.

Step 1 Using the optimal department-to-floor assignments obtained from FAF, on each floor arbitrarily sequence the departments in increasing order of the department indices. Let this sequence be s^0. Set $s = s^0$, $I = 1$, $i = 1$ and $\boldsymbol{a}$ consistent with s^0. (The exact values of $\boldsymbol{a}$ are not critical as long as the floor numbers are observed and the variable components are consistent with s^0.)

Step 2 For the current layout vector, compute the cost of the layout, say, $f(s)$, from (1) and (14).

Step 3 a Sample a vector of random variables $\boldsymbol{b}$, where each element of $\boldsymbol{b}$ is distributed $U(0,1)$.

b For each element i, if $b(i) \leq \beta$, sample a random number, say, a, where a is distributed $U(0,1)$. Set the new ith element of $\boldsymbol{a}$ equal $y_i + a$. Sort $\boldsymbol{a}$ to determine the new layout vector. Set this vector to s', construct the corresponding layout, and go to step 3c.

c Evaluate the resulting change in layout cost, $\Delta f = f(s) - f(s')$. If $\Delta f \leq 0$, then go to step 3d; otherwise go to step 3e.

d Select a random variable $x \sim U(0,1)$. If

$$x < P(\Delta f) \equiv \exp(\Delta f / t_i),$$

then go to step 3e; otherwise go to step 3a.

Step 3 e Accept the candidate vector, set $s = s'$ and $f(s) = f(s')$. If $f(s) < f(s^*)$, then set $s^* = s$, $f(s^*) = f(s)$ and $I = i$. If e layout vectors have been accepted, go to step 4; otherwise go to step 3a.

Step 4 If equilibrium has not been reached at temperature t_i, i.e., if $|\overline{f}_e - \overline{f}'_e|/\overline{f}'_e \geq \epsilon$, then go to step 3a; otherwise, set $i = i + 1$ and $t_i = t_0(0.8)^{i-1}$. If $(i - I) < N$ go to step 5; otherwise **STOP**, the maximum number of successive non-improving temperature settings has been reached.

Step 5 If the total number of epochs is less than M then go to step 3a; otherwise, **STOP.**

The final layout obtained from SABASS defines the centroid locations of the departments. We use these centroid locations and other data to solve the third stage problem, i.e., the LLAP, to determine which lifts should be open and how to allocate each vertical flow to an open lift.

6 Lift Location-Allocation Problem

In the first two stages, to determine the layout, we assumed that all potential or existing lift locations designated by the user are "open" and uncapacitated. Since these conditions do not hold in general, we must decide which lift sites to open, and assign each vertical flow to one of the open lifts without exceeding that lift's throughput capacity.

We assume that each (vertical) flow must be assigned to exactly one lift. Splitting some or all of the flows among multiple lifts may yield better lift capacity utilizations and it may seem more flexible. However, from a practical viewpoint it would be undesirable since its effective implementation requires well-defined rules that govern "lift selection" amongst multiple lifts each time material is moved. (Even simple rules such as picking the lift with minimum queue length may force the material handler to inspect each lift.) Furthermore, enforcing such rules may be impossible or impractical. We also assume that each lift location contains exactly one lift. That is, we do not consider the case where two or more lifts are grouped in a single location with a single queue forming in front of the lifts.

Lastly, we assume that the loads associated with each flow arrive at a lift, one at a time, in a Poisson fashion that can be specified by a mean arrival rate and a routing matrix, which indicates the probability that a load picked up at floor k needs to be delivered to floor g. (The routing matrix can be easily constructed from the flows assigned to a particular lift.) All the loads are served one at a time on a first-come-first-served (FCFS) basis at each lift. (Of course, one "load" may consist of several containers that are handled together.) The distance between the floors and the lift travel speed(s), as well as any load pick-up and deposit times, are given and fixed.

The overall layout cost is based on three components: the amortized cost of the (open)

lifts, the annual cost of travel between departments on the same floor, and the annual cost of travel between departments on different floors. Consider two departments, i and j, located on different floors. The cost to travel from department i to department j can be expressed as the sum of the cost to travel horizontally from department i to the lift, the cost associated with waiting for the lift, the cost to travel vertically on the lift, and the cost to travel horizontally from the lift to department j. The cost to travel vertically on the lifts may be ignored since it is constant. (Recall that the layout is fixed.) Likewise, the cost to travel horizontally between departments on the same floor may also be ignored. Hence, the three terms in the LLAP objective function are the (amortized) cost of opening lifts, the cost of traveling horizontally between the departments and the lifts, and the cost of waiting at the lifts.

For ease of exposition, let us re-index the flows ($m = 1, \ldots, M$) to include only the *vertical* flows. In addition to the parameters defined previously, the following must be given as data:

1. The magnitude of the mth vertical flow is given by f_m, where $f_m > 0 \ \forall m$.
2. The cost to travel one horizontal distance unit for flow m is c_m^H.
3. The amortized cost to open lift ℓ is C_ℓ^O.
4. The cost to wait for lift ℓ is C_ℓ^W per time unit. (For ease of exposition, we assume that the waiting time cost is independent of flow type. It is straightforward to relax this assumption as long as flows with higher waiting-time costs do not receive higher priority service from a lift.)

Let us next define two additional terms. The horizontal travel distance required if flow m is assigned to lift ℓ, $d_{m\ell}^H$, is given by

$$d_{m\ell}^H = d_{i(m)\ell}^H + d_{j(m)\ell}^H. \tag{16}$$

Therefore, the "horizontal cost" to assign flow m to lift ℓ, C_{ml}^H is given by

$$C_{ml}^H = c_m^H d_{m\ell}^H. \tag{17}$$

6.1 LLAP Formulation

Assuming that our objective is to minimize the total cost, we obtain the following formulation of the LLAP. Let ρ_ℓ and W_ℓ denote the expected utilization of lift ℓ and the expected waiting time at lift ℓ, respectively.

$$\min \quad \sum_{\ell} C_{\ell}^{O} u_{\ell} + \sum_{m} \sum_{\ell} C_{m\ell}^{H} z_{m\ell} + \sum_{\ell} C_{\ell}^{W} W_{\ell} \tag{18}$$

$$\text{s.t.} \quad \sum_{\ell} z_{m\ell} = 1 \qquad \forall m \tag{19}$$

$$\rho_{\ell} < u_{\ell} \qquad \forall \ell \tag{20}$$

$$z_{m\ell} \leq u_{\ell} \qquad \forall m, \ell \tag{21}$$

$$\text{where} \quad \rho_{\ell} = r(\boldsymbol{z}) \qquad \forall \ell \tag{22}$$

$$W_{\ell} = s(\rho_{\ell}) \qquad \forall \ell \tag{23}$$

$$\text{and} \quad u_{\ell} = \begin{cases} 1 & \text{if lift } \ell \text{ is opened,} \\ 0 & \text{otherwise} \end{cases} \tag{24}$$

$$z_{m\ell} = \begin{cases} 1 & \text{if } m\text{th flow is assigned to lift } \ell, \\ 0 & \text{otherwise} \end{cases} \tag{25}$$

Constraint (19) ensures that each flow is assigned to exactly one lift. Constraint (20) ensures that the lift utilization is less than one for each open lift, and equal to zero for each unopened lift. Constraint (21) ensures that flows are assigned only to open lifts. Of course, the expected utilization of a lift is a function of the flows assigned to that lift (see (22)), and likewise, the expected waiting time at a lift is a function of the lift utilization (see (23)).

Recall that the decision variable, $z_{m\ell}$, equals one if the mth flow is assigned to lift ℓ, and zero otherwise. The utilization of lift ℓ, ρ_{ℓ}, is given by

$$\rho_{\ell} = \sum_{m} f_m z_{m\ell} E[S_{\ell}], \tag{26}$$

where $E[S_{\ell}]$ represents the expected service time per load for lift ℓ. The expected service time is composed of the expected empty lift travel time to pick up the next load, $E[E_{\ell}]$, and the expected loaded lift travel time, $E[L_{\ell}]$, to deliver it to the appropriate floor. That is,

$$E[S_{\ell}] = E[E_{\ell}] + E[L_{\ell}]. \tag{27}$$

For a given set of flows assigned to lift ℓ, it is straightforward to determine $E[L_{\ell}]$. For empty lift travel, on the other hand, we note that the amount of empty travel required to pick up a particular load is a random variable (since the location of the lift at any point in time is a stochastic process); its expected value is given by:

$$E[E_{\ell}] = \sum_{m} p_{m\ell} E[E_{m\ell}], \tag{28}$$

where $p_{m\ell}$ represents the probability that the load to be serviced (by lift ℓ) belongs to flow m, and $E[E_{m\ell}]$ denotes the expected empty travel time for lift ℓ to pick up a load

associated with flow m. Due to competing exponential interarrival processes, $p_{m\ell}$ is given by:

$$p_{m\ell} = \frac{z_{m\ell} f_m}{\sum_{m'} z_{m'\ell} f_{m'}}. \tag{29}$$

Since (29) is a non-linear function of the decision variables, the constraints in set (20) are non-linear. In a similar manner, the waiting time function, W_ℓ, is also non-linear, which leads to a non-linear objective function (see(18)).

The above result also points to a significant difference between loaded and empty lift travel, and the role they play in the LLAP formulation. The amount of *loaded* travel imposed on a lift by a particular flow does not depend on other flows assigned to that lift. Therefore, it can be computed *a priori*. However, as evidenced by (28) and (29), the amount of *empty* travel imposed on a lift by a particular flow depends on other flows assigned to that lift. For example, suppose flows m and $m+1$ both travel from floor k to g, while flow $m+2$ travels from floor g to k. Assigning flows m and $m+2$ to the same lift will generate less empty lift travel than assigning flows m and $m+1$. Due to such "interaction" among the flows, determining the expected lift utilization (which includes empty travel) is not straightforward.

Since we have not derived $E[E_{m\ell}]$ and W_ℓ in closed-form, the LLAP formulation (defined by (18) - (25)) is incomplete. However, given Poisson arrivals and FCFS service, for a single-device material handling system such as a lift, Cho [4] presents an *M/G/1* queueing model. Using his model (which is presented in the Appendix), one may express the LLAP formulation in closed-form. However, due to the non-linearities involved (both in the objective function and one of the constraint sets), the resulting model would be difficult to solve. Adding to the difficulty is the observation that the cost associated with an open lift is based on a fixed charge of C_ℓ^O and a convex function of the total flow assigned to the lift (W_ℓ). Thus, there are both economies and diseconomies of scale associated with the amount of flow assigned to a lift.

However, we need to solve the LLAP since it captures one of the basic tradeoffs of material handling in a multi-floor facility with a given layout. That is, assigning the lift that is closer in travel distance versus the lift that has shorter expected waiting times. The lift that is closer in travel distance may be highly utilized and thus have a longer average waiting time than another lift that may be farther in travel distance, but with a shorter average waiting time. In the next section we present a heuristic algorithm to solve the LLAP.

6.2 Solution Strategy

The complete model with expected waiting times is heuristically solved with a simulated-annealing based algorithm, which is similar to the simulated-annealing based algorithm developed in §5 for the second stage problem. Like SABASS, by starting with an ini-

tial solution and generating candidate solutions in a prescribed random fashion, we seek improvements to the initial solution.

A candidate solution is represented by a vector of lift indices that we denote by $\mathcal{L}$. The mth element of $\mathcal{L}$, $\mathcal{L}_m$, denotes the lift index to which flow m is assigned. For example, for $M = 8$ and $L = 4$, if the values of $\mathcal{L}$ are given by 1–3–1–4–3–1–3–4, it indicates that flows 1, 3, and 6 are assigned to lift 1, flows 2, 5 and 7 are assigned to lift 3, flows 4 and 8 are assigned to lift 4, and lift 2 is not open. Thus, each element of $\mathcal{L}$ is an integer number between one and the number of potential lift sites (L).

To check a candidate solution for feasibility, we use the analytical model presented in the Appendix and compute the expected utilization of each (open) lift. If the utilization of any lift exceeds a user-defined upper limit (say, 0.90 or less), then we declare the candidate solution infeasible and generate another one. If the candidate solution is feasible, then we use the model presented in the Appendix to compute the expected waiting time at each lift. Subsequently, we evaluate the objective function (given by 18) to determine the cost of the candidate solution.

Candidate solutions are generated with a procedure similar to the one used in SABASS. In addition to its lift assignment ($\mathcal{L}_m$), each flow has a variable associated with it between zero and one. This variable, $b(m)$, serves the same purpose as $b(i)$ in SABASS. Starting with the first flow, we assign $b(1)$ to a random number sampled uniformly between zero and one. If $b(1)$ is less than a user-set critical value, β, then we sample an integer number between 1 and L to assign to $\mathcal{L}_1$, which represents the new lift assignment for the first flow. Otherwise, the lift assigned to the first flow remains the same. To generate a candidate solution, we repeat the above procedure for all the flows in the current solution.

Lastly, the initial temperature (t_0) is determined as a function of the initial objective function value (z_0). Using trial-and-error, we obtained the following function:

$$t_0 = z_0(1/10). \tag{30}$$

The approach used to control the overall annealing schedule, including temperature reduction and establishing equilibrium, is the same as the approach used in SABASS. We now present the necessary notation and the simulated annealing algorithm.

6.3 Notation and Algorithm

We use the following notation in the simulated-annealing based algorithm developed for the LLAP. Let

$\mathcal{L}^0$ = initial assignment of flows to lifts.

$\mathcal{L}^*$ = current best assignment of flows to lifts; it represents the lowest cost assignment identified by the algorithm.

$\mathcal{L}$ = the current assignment of flows to lifts.

$\mathcal{L}'$ = the candidate assignment of flows to lifts.

T = $\{t_1, t_2, t_3, \ldots\}$ — a set of annealing schedule temperatures, where $t_i = t_0(0.8)^{i-1}, \forall i > 1$.

t_0 = initial temperature.

e = epoch length — fixed number of candidate assignments that will be accepted during each epoch.

$f_j(\mathcal{L})$ = objective value of the jth accepted candidate assignment, $\mathcal{L}$.

$\overline{f}_e$ = $\left(\sum_{j=1}^{e} f_j(\mathcal{L})\right)/e$ — the mean objective function value of an epoch with e accepted candidate assignments.

$\overline{f}'_e$ = the overall mean of objective function values for the assignments accepted during the epochs previous to the current one for a given temperature.

ϵ_i = an error constant used to determine whether the system is in equilibrium at temperature i.

$\boldsymbol{b}$ = a vector of (0,1) random variables that determines if the lift assignments will change.

β = critical value to determine whether a new lift assignment is made for a flow, $\beta \in (0,1)$.

M = a number that is specified *a priori* to control the maximum number of epochs considered.

I = a counter to record the temperature setting which produced the current best assignment, $\mathcal{L}^*$.

N = the maximum number of successive temperature settings that do not produce a new $\mathcal{L}^*$, $I \leq N$.

The variables $t_1, e, \epsilon, \beta, M$ and N are control variables that are set by the user. The settings used for the control variables will be described in subsequent sections. A general outline of the algorithm to solve the LLAP by simulated annealing (i.e., LLAPSA) follows. For an initial temperature, t_0:

Step 1 Randomly generate an initial flow-to-lift assignment, $\mathcal{L}^0$, until the assignment is feasible. Set $\mathcal{L} = \mathcal{L}^0$, $I = 1$, and $i = 1$.

Step 2 Compute the cost of the current assignment, $f(\mathcal{L})$, from equation (18).

Step 3 a Select a vector of random variables $\boldsymbol{b}$, where each element of $\boldsymbol{b}$ is distributed $U(0,1)$.

b For each element m of $\boldsymbol{b} \leq \beta$, generate a new lift assignment for the flow, $\mathcal{L}_m$. Determine whether the assignment $\mathcal{L}$ is feasible. If feasible, then set the assignment to $\mathcal{L}'$ and go to step 3c; otherwise go to step 3a.

Step 3 c Evaluate the resulting change in cost, $\Delta f = f(\mathcal{L}) - f(\mathcal{L}')$. If $\Delta f \leq 0$, then go to step 3d; otherwise go to step 3e.

d Select a random variable $x \sim U(0,1)$. If

$$x < P(\Delta f) \equiv \exp(\Delta f / t_i),$$

then go to step 3e; otherwise go to step 3a.

e Accept the candidate assignment, set $\mathcal{L} = \mathcal{L}'$ and $f(\mathcal{L}) = f(\mathcal{L}')$. If $f(\mathcal{L}) < f(\mathcal{L}^*)$, then set $\mathcal{L}^* = \mathcal{L}$, $f(\mathcal{L}^*) = f(\mathcal{L})$, and $I = i$. If e assignments have been accepted go to step 4; otherwise go to step 3a.

Step 4 If equilibrium has not been reached at temperature t_i, i.e., if $|\overline{f}_e - \overline{f}'_e|/\overline{f}'_e \geq \epsilon$, then go to step 3a; otherwise, set $i = i + 1$ and $t_i = t_0(0.8)^{i-1}$. If $(i - I) < N$ go to step 5; otherwise **STOP**, the maximum succession of non- improving temperatures has been reached.

Step 5 If the total number of epochs is less than M, go to step 3a; otherwise, **STOP.**

It is worth noting that the above simulated-annealing based algorithm could be modified to consider alternate queueing models. In its current form, LLAPSA uses an analytical model which assumes FCFS service. As shown in [4], when the lift utilization is high, the FCFS policy is inferior to the *modified first-come-first-served* (MOD FCFS) policy. (With MOD FCFS, when the lift delivers a load to, say, floor k, it first inspects floor k for outgoing loads. If an outgoing load is found, it is served first regardless of its arrival time. Otherwise, the lift serves the "oldest" waiting load in the system as before.) However, in many cases, in order to maintain reasonable queue sizes, lift utilizations are likely to be less than 0.80 [16]. Thus, a FCFS policy is likely to result in only slightly larger expected waiting times and lift utilizations. (The impact of alternate service policies also depends on the particular set of flows assigned to a lift.) In any case, using alternate queueing models would not alter the mechanics of the above simulated annealing heuristic.

With the development of a construction-type layout algorithm for the first two stages, and an algorithm to solve the LLAP in the third stage, we are now in a position to construct a *final solution*, which not only specifies the multi-floor layout, but also the lift system configuration. In the next section we examine the performance of the individual algorithms, as well as the quality of the final solutions we obtained.

7 Results

In this section we present three sets of results. First, we compare the layouts we obtain from the first two stages with those layouts obtained with SABLE [17]. Second, for given test layouts, we compare the results obtained from the third-stage algorithm alone with

the third-stage optimal solutions. Third, for five small problems, we compare the results of our three-stage approach (i.e., the final solution) with the globally-optimal solution, which is obtained by simultaneously considering the layout and the lift location-allocation problems.

7.1 Layout Results

Although SABLE is an improvement-type approach, it has been shown to generally produce the lowest-cost layouts for multi-floor layout problems [17]. Also, results obtained from SABLE would be comparable to ours in that, unlike other multi-floor layout algorithms, SABLE accepts multiple lift locations and unequal departmental area requirements without splitting departments across floors. (Although MULTIPLE [3] has the same attributes, it is outperformed by SABLE.)

For convenience, we will refer to the layout algorithm that is obtained by implementing the first two stages of our approach as STRING. The following values were used for the control settings shown in parenthesis: 30 (e), 0.25 (ϵ_1), 0.05 ($\epsilon_i, i \geq 2$), 0.10 (β), 37 (M) and 5 (N). The initial temperature was specified as a function of the initial layout cost and is obtained from [17].

STRING solutions compare quite favorably with SABLE solutions on the test problems presented in [17]. (The data sets are referenced by the number of departments, the number of floors, and an index to further differentiate similar data sets.) The final STRING solution for each of the seven data sets is presented in Figure 2 along with the minimum, average and maximum values achieved with SABLE over 10 initial layouts. For most data sets, STRING achieves final objective values that are less than or equal to the minimum solution values obtained with SABLE. Note that, for any data set, the optimal layout *for the given spacefilling curve* may be determined by considering all possible department sequences. In [17], the "optimal" layouts are reported for data sets 11-2-1, 11-2-2, and 12-3. STRING achieves this optimal cost in two of the three data sets. Also note that, for three of the four 21-department data sets and the 40-department data set, the STRING cost is less than the minimum cost obtained with SABLE over ten initial layouts.

A runtime comparison of the two algorithms is not straightforward since STRING and SABLE were implemented on different computer platforms. On a 25 MHz 386 DOS-based personal computer (PC), the average runtime of SABLE with one initial layout was 250.43 seconds for the 21-department problems. The first stage of STRING was implemented on an IBM RISC/6000 320H workstation (using OSL) while the second stage was implemented on the above PC. Although the total runtime is the sum of the stage-one and stage-two solution times, since these solutions were obtained on different platforms we cannot simply add them. To facilitate a more meaningful comparison, we developed a simple benchmark program to estimate the speed ratio between the RISC/6000 and the above PC. Our test results indicate that the RISC/6000 is approximately 7.2 times faster

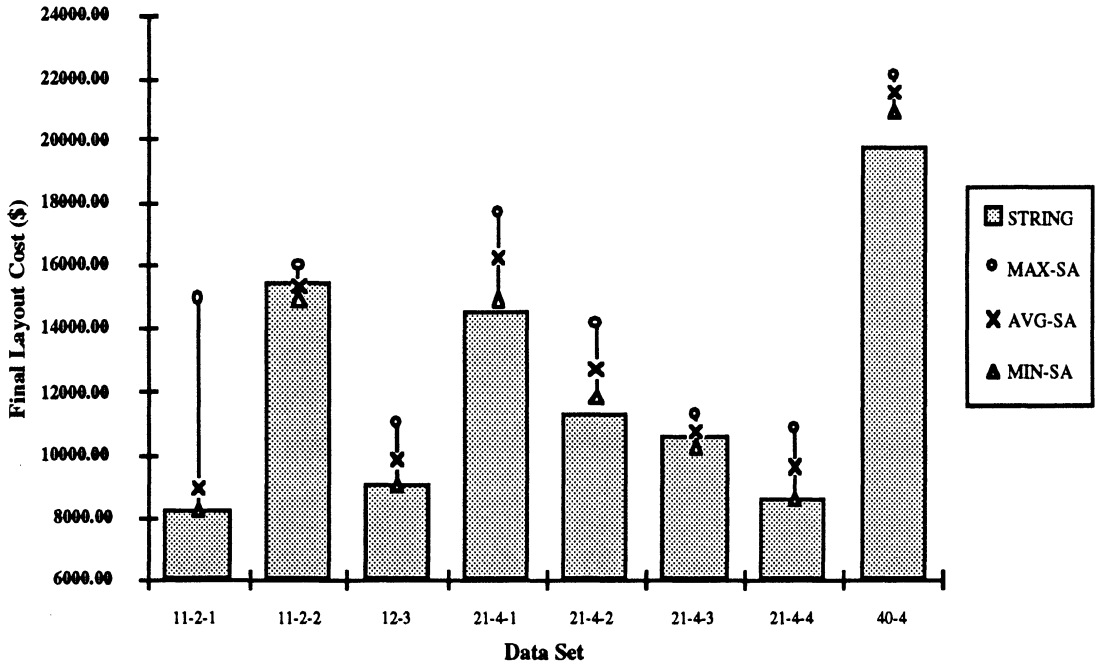


Figure 2: A Comparison of STRING and SABLE.

than the PC.

Using the above conversion factor, the STRING stage-wise runtimes and the SABLE runtimes (expressed in CPU seconds) are presented in Table 1. Excluding data set 40-4, the STRING runtimes are between 0.5 and 5.6 times as long as the SABLE runtimes depending on the data set. (Recall that the SABLE runtimes are based on one initial layout.) The above variation is due to the solution of the first-stage integer program in STRING. Considering the significance of the facility layout problem, and the frequency with which it is solved, the Table 1 runtimes (obtained on a relatively slow PC) are within reason. Obviously, the difficulty of solving FAF for the 40-department problem may require the development of a heuristic. (With the heuristic presented in [16], the runtime of the two-stage layout construction algorithm is approximately equal to the SABLE runtime for one initial layout.)

The two-stage approach fundamentally assumes that the unit cost of vertical travel is greater than the unit cost of horizontal travel. Therefore, we need to examine the sensitivity of STRING's performance to the ratio of vertical to horizontal unit travel costs, i.e., the ratio of c^V to c^H, or the "V/H Ratio." (For the data sets shown in Figure 2, the V/H ratio is equal to 5:1.) As the V/H ratio is increased, we would expect STRING to perform better relative to SABLE. To illustrate this point, the 21-department data sets were solved with a high V/H Ratio of 20:1. The results are shown in Figure 3. As expected, STRING performs very well. The 21-department data sets were also solved

Table 1: Comparison of STRING and SABLE Runtimes.

Data Set	11-2-1	11-2-2	12-3	21-4-1	21-4-2	21-4-3	21-4-4	40-4
Actual Stage 1 Rtime	1.05	1.64	2.54	80.25	13.57	205.50	7.09	9876.4
Est. Stage 1 Rtime	7.56	11.81	18.23	577.80	97.70	1479.60	51.05	71110.1
Actual Stage 2 Rtime	6.48	7.27	6.42	67.33	55.81	81.23	53.66	189.2
STRING Total Rtime	14.04	19.08	24.65	645.13	153.51	1560.83	104.71	71299.3
SABLE Rtime	17.78	14.26	20.37	283.52	249.57	277.13	191.48	2174.8

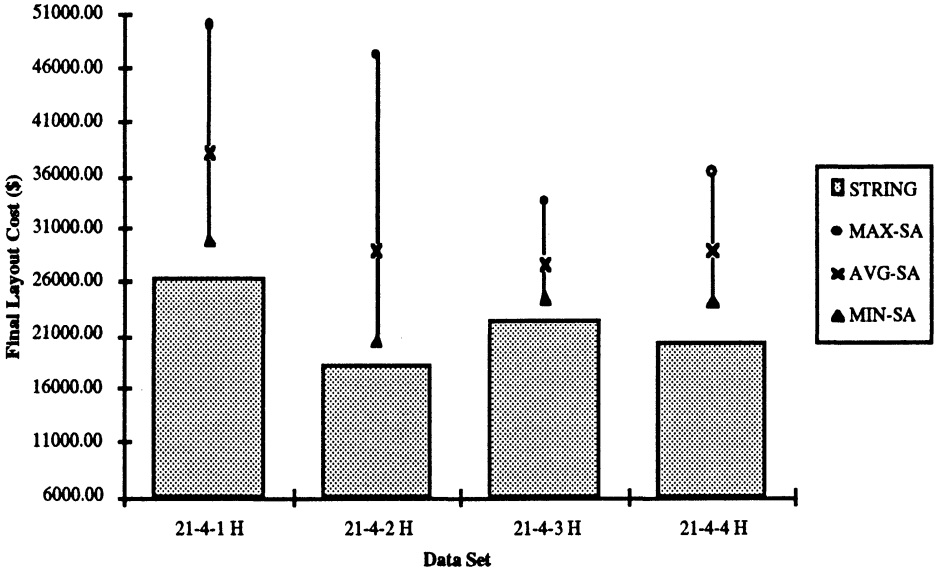


Figure 3: A Comparison of STRING and SABLE on Data Sets with a High V/H Ratio.

with a low V/H Ratio of 1:1. The results are shown in Figure 4, where STRING still performs well. The value of the V/H Ratio for a particular application depends on the products handled and the material handling technology used in the facility.

7.2 Lift Configuration Results

In this section we compare the solutions obtained from LLAPSA with the third-stage optimal solutions, which are obtained by considering all possible allocations of flows-to-lifts. The layout problems used in §7.1 were used to construct the LLAP test problems by selecting the minimum-cost layout obtained between SABLE and STRING.

The unit horizontal and vertical travel costs, and the potential lift locations, are

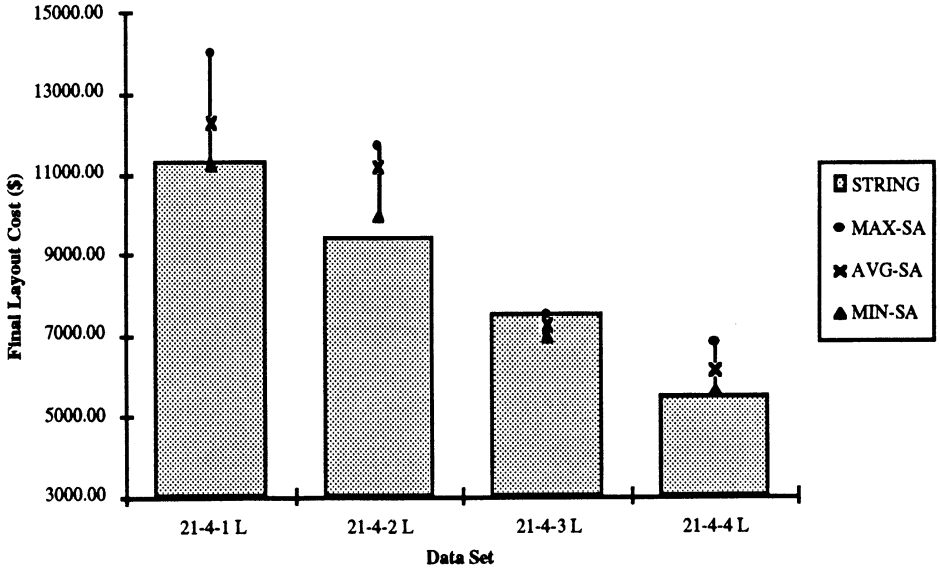


Figure 4: A Comparison of STRING and SABLE on Data Sets with a Low V/H Ratio.

specified in the layout test problems. (The department centroids are obtained from the minimum-cost layout itself; recall that the layout is now assumed to be fixed.) Additional parameter values required for the LLAP test problems (i.e., the waiting cost per time unit, the lift speed, and the amortized lift cost) were selected as follows. Since the waiting time represents an opportunity cost for material handling resources (i.e., the material handler and/or the capital investment in equipment), the cost to wait for a lift was set equal to \$40 per hour. To increase the feasible solution space, the lift speed was set equal to a value that is likely to make it possible to assign all the flows to one lift without exceeding its throughput capacity. Based on trial-and-error, we arrived at the following function: lift speed = (number of floors − 0.4) × (loaded vertical travel distance required if all flows are assigned to one lift). For example, if the loaded vertical travel distance required in a two-floor facility for one hour's production is 940 feet, we would set the lift speed equal to 1500 feet per hour (or 25 feet per minute).

While the waiting cost and lift speed are independent of the layout problem parameters, the lift cost itself is not. The vertical unit handling costs that are used in the layout problem (i.e., the c^V values) depend on the type of vertical handling equipment used, which in turn dictates the lift opening cost for the LLAP. Therefore, the lift cost must be selected carefully in order to avoid unrealistic test problems. Let x be a conversion factor which equalizes the horizontal cost of handling one unit of flow between the opposite corners in a floor to the vertical cost of handling one unit of flow between two adjacent floors. That is, if d^{max} represents the maximum horizontal travel distance in a floor, e represents the vertical distance between adjacent floors, and we assume horizontal and

Table 2: The Data Specifications for the LLAP Test Problems.

Data Set	num flows	num lifts	cost to wait	lift speed	lift cost
11-2-1	6	2	40.0	1500	32.9
11-2-2	10	2	40.0	1700	210.1
12-3	10	2	40.0	1950	120.0
21-4-1	12	2	40.0	2950	1050.0
21-4-2	13	2	40.0	1700	600.0
21-4-3	13	2	40.0	3500	1235.0
21-4-4	12	2	40.0	2700	900.0
40-4	27	3	40.0	3300	500.0

vertical unit costs (c^H and c^V, respectively) to be equal between every department pair, then x should satisfy the following relationship:

$$(d^{max}) \times (c^H) = (e) \times (c^V) \times (x). \tag{31}$$

To determine the amortized lift cost, we multiply the above x value with the total vertical flow in the facility.

For example, in data set 11-2-1, the grid size is 2.5 distance units, c^H is equal to \$1.0, and the maximum (rectilinear) distance traveled within a floor is 7.0 grids (since the layout is 6 by 3 grids and we take the centroid of each grid). Thus, d^{max} is equal to 17.5 distance units. The distance between adjacent floors is 10.0 distance units, and c^V is equal to \$5.0. Hence, using (31) we obtain $x = 0.35$. Since the total vertical flow is 94 units per time period, the lift cost for data set 11-2-1 is set equal to \$32.9. Such a value represents the amortized lift cost over a time period (which corresponds to the time period of the flow data).

To generate LLAP test problems we used data sets 11-2-1, 11-2-2, 12-3, 21-4-1, 21-4-2, 21-4-3, 21-4-4, and 40-4. The LLAP parameters are displayed in Table 2, where lift speed is expressed in distance units/time unit. The following values were used for the control settings (indicated in parenthesis) of LLAPSA: 100 (e), 0.20 (ϵ_1), 0.05 ($\epsilon_i, i \geq 2$), 0.20 (β), 10 (M), and 3 (N). The initial temperature was specified according to (30). A single, randomly generated initial solution was used for each data set.

Results concerning the performance of LLAPSA relative to the optimal solution are presented in Table 3, where (H Val) and (Opt Val) represent the heuristic and optimal solution values, respectively. The runtimes (expressed in PC CPU seconds) are also shown in Table 3 under "H RTime" and "Opt RTime" for the heuristic and optimal solutions, respectively. The column labeled "Num Lifts Opt" shows the number of lifts that are open in the optimal solution out of the total number of potential lifts. (The optimal solution to data set 40-4 is not available.) Note that, with the potential exception of data

Table 3: Comparison of Heuristic and Optimal Solutions for LLAP Data Sets.

Data Set	H Val	Opt Val	H RTime	Opt RTime	Num Lifts Opt
11-2-1	1116.78	1116.78	0.40	0.22	2/2
11-2-2	1146.66	1146.66	0.65	2.09	2/2
12-3	590.88	590.88	1.15	3.84	2/2
21-4-1	6534.77	6534.77	2.42	26.91	2/2
21-4-2	3457.79	3457.79	2.75	54.54	2/2
21-4-3	4511.68	4511.68	2.57	53.72	2/2
21-4-4	2951.21	2951.21	2.57	53.83	1/2
40	7231.99	N/A	40.37	N/A	N/A

Table 4: Comparison of Heuristic and Optimal Solutions for Modified LLAP Data Sets.

Data Set	H Val	Opt Val	H RTime	Opt RTime	Num Lifts Opt
11-2-1	1052.37	1052.37	4.06	1.87	3/3
11-2-2	1146.66	1146.66	8.07	142.87	2/3
12-3	493.00	493.00	11.37	304.62	1/3
21-4-1	6534.77	6534.77	20.49	5216.38	2/3
21-4-2	3457.79	3457.79	22.64	15745.61	2/3
21-4-3	4511.68	4511.68	27.52	15660.00	2/3
21-4-4	2951.21	2951.21	18.34	15740.27	1/3

set 40-4, LLAPSA obtained the optimal solution for all the test problems.

In Table 3, since each potential lift was opened in the optimal solution to almost all of the test problems, additional test problems were generated. Disregarding data set 40-4, one potential lift (with the specifications listed in Table 2) was added to each of the test problems. These new test problems were then solved with LLAPSA. The results are presented in Table 4 where we note that an additional potential lift site did not always affect the optimal solution, although it did increase the computational effort of the algorithms. As before, with very reasonable runtimes, LLAPSA obtained the optimal solution to every test problem. The optimal solutions to the LLAP test problems and the lift locations are presented in [16].

7.3 Layout and Lift Configuration Results

In this section we evaluate the cost of the final solution (obtained from the three-stage heuristic approach) against the cost of the globally-optimal solution, which was obtained by considering all possible layouts, and for each layout, all possible lift configurations

(i.e., all possible flow-to-lift assignments). Since such a brute-force method is quite time consuming, we were able to obtain the global optimal for only five relatively small test problems. The first two problems have nine departments and two floors, the third problem has nine departments and three floors, and the fourth and fifth problems have 10 departments and two floors. Each problem has one fixed department to represent the receiving/shipping department.

In preparing the data for the above problems, we conducted a few experiments in order to guarantee that the optimal solution to the (third stage) LLAP did *not* use the same lifts that were used in the layout solution obtained at the end of the second stage. In some cases this meant specifying high lift costs. Without such a measure, the test problems would be less interesting (or challenging) since the three-stage heuristic is unlikely to produce non-optimal solutions if all the lifts used by the layout algorithm are still open after the lift location-allocation problem has been solved. The test problems are referenced by the number of departments, the number of floors, and an index to further differentiate similar test problems. For example, the first two test problems with nine departments and two floors are denoted as problems 9-2-1 and 9-2-2, and so on. (All the data for the five test problems are presented in [16].)

In all the test problems, we used STRING to solve the layout problem. (Recall that, in the layout problem, we assume that all existing/potential lifts are open and that each lift has unlimited capacity. As a result, each flow that involves vertical travel is assigned to the lift that minimizes total horizontal travel to and from the lift.) For all test problems except problem 10-2-1, STRING obtained a layout which was "optimal" for the first two stages. The resulting layouts were subsequently used as input to LLAPSA. Test results indicate that in all five problems, LLAPSA obtained the optimal lift configuration for the given layout obtained by STRING.

The globally-optimal solution, on the other hand, is obtained by implicitly considering all possible solutions. That is, for every possible layout sequence, the layout cost is calculated assuming that each vertical flow is assigned to the lift which minimizes the horizontal travel to and from the lift. For each layout sequence with one or more vertical flows, a lower bound on the total cost is computed as the sum of the layout cost (computed above) and the cost of one lift. (Of course, with realistic flow data, each layout sequence will have two or more vertical flows.) If the lower bound on the total cost is less than or equal to the incumbent cost, then every possible allocation of vertical flows to lifts is considered to determine the total cost of the layout sequence. After considering all possible layout sequences, the resulting incumbent solution will be the globally-optimal solution. We will refer to the above algorithm as GLOBAL.

The results of the comparison between the three-stage heuristic solutions and the solutions obtained from GLOBAL are presented in Table 5. The runtimes for the two algorithms are also shown in Table 5. (The solutions were obtained on the PC or converted to PC-equivalent runtimes.) The results indicate that for relatively small problems, the three-stage heuristic we propose produces globally-optimal or near-globally-optimal

Table 5: A Comparison Between the Three-Stage Heuristic and the Globally-Optimal Solutions.

Data Set	3-Stage Cost	3-Stage Rtime	GLOBAL Cost	GLOBAL Rtime
9-2-1	2465.56	23 seconds	2458.63	30 minutes
9-2-2	2116.95	25 seconds	2116.95	30 minutes
9-3	1228.53	43 seconds	1190.58	33 hours
10-2-1	1703.77	23 seconds	1673.77	21 hours
10-2-2	3124.64	24 seconds	3124.64	60 hours

solutions. For data set 9-3 (which produced the largest gap between the two costs) the three-stage heuristic solution is only 3.2% above the global optimum. To test larger problems with GLOBAL would require a faster computer since the runtime increases exponentially with the number of departments, lifts, and vertical flows.

We recognize that the three-stage approach may not perform as well on larger problems. However, the layout problem may be re-solved with an algorithm (such as SABASS or SABLE) after the lift location-allocation problem has been solved and vice versa. That is, one can iterate between SABASS and LLAPSA until the number, location and the flow-to-lift assignments do not change from one iteration to the next. While such an iterative scheme is not guaranteed to locate the globally-optimal solution, it is likely to improve the performance of the three-stage heuristic in large problems with many departments and more than 3 or 4 lifts. Given the runtimes shown in Table 5 for the three-stage heuristic, an iterative scheme for larger problems seems entirely reasonable.

8 Conclusions

In this paper we present a three-stage heuristic algorithm to solve the multiple floor facility layout problem. The heuristic is based primarily on mathematical-programming and simulated-annealing. To our knowledge, our approach represents the most comprehensive treatment of this problem since we do not make restrictive assumptions about building shape, department areas, or the number and location of lifts. We integrate the spacefilling curve representation presented in [3] into a construction-type layout algorithm and also solve the LLAP. Both the layout algorithm and the LLAP algorithm performed well when compared to previous algorithms or their respective optimal solutions. Furthermore, the test problems presented in §7.3 indicate that, on relatively small problems, the three-stage approach based on sequentially solving the layout problem followed by the lift location-allocation problem, compares favorably with the global optimal which is obtained by solving the two problems simultaneously.

Table 6: A Comparison Between Single- and Multi-Floor Layout Costs with Two V/H Ratios.

Data Set	Single	Multi 1:1	Multi 5:1
21-1	9846.00	11241.00	14505.50
21-2	9433.33	9399.00	11315.00
21-3	7606.00	6778.83	10263.50
21-4	6754.00	5466.33	8556.33

In §1 we remarked that it may be possible to achieve lower *layout costs* with a multi-floor facility than with a single-floor facility (especially if the vertical unit costs are approximately equal to the horizontal unit costs). To demonstrate this point, the four 21-department 4-floor test problems presented earlier were "converted" into single-floor test problems with identical area and flow data. (The spacefilling curve shown in [16] was used with SABLE for all the single-floor problems.) The same building length-to-width ratio and the same 10 initial department sequences were used for both the multi- and single-floor problems.

The results of this small experiment are shown in Table 6, where the best known layout cost for both the single-floor and the multi-floor problem are presented for a V/H Ratio of 1:1 and 5:1. Note that, with a V/H Ratio of 1:1, in three out of the four data sets a lower layout cost was achieved with a multi-floor building. In the above data sets, the length (width) of each building is only 8 times (4.8 times) the distance between adjacent floors. In general, we would expect the length and width of the facility to be much greater than the distance between adjacent floors. In such cases, a multi-floor facility is likely to yield a lower-cost layout than a single-floor facility unless the V/H Ratio is very high.

Additional work is required to compare single and multiple floor production facilities. The final determination must be made on the basis of feasibility and economic factors including layout costs, construction costs, land costs/availability, energy costs, and maintenance costs. In this paper we focused only on the layout costs.

ACKNOWLEDGMENT

The authors would like to thank Derek Holmes for his assistance in developing the computer code that was used to solve the FAF presented in §4.

Appendix

Determining Lift Utilization and Waiting Times

The expected lift utilization and the expected waiting times are determined with the M/G/1 model developed by Cho [4]. The loads associated with each flow assigned to the lift arrive at the lift one at a time, and they are served, one at a time, on a first-come first-served (FCFS) basis. Although the model presented in [4] can handle variable travel times, we assume that all travel times are deterministic. We stress that in this appendix we are not concerned with how flows are assigned to lifts. Rather, given the flows assigned to a particular lift, we would like to determine the expected lift utilization (ρ), and the expected waiting time per load (W). We use the following notation (which is a simplified version of Cho's notation):

K = the number of floors.

λ_{kg} = the arrival rate of loads at floor k that must travel to floor g; that is, $\lambda_{kg} = \sum_{m=1}^{M} f_m : i(m) = k,\ j(m) = g.$

λ_k = the arrival rate of loads from floor k that must travel to all other floors; that is, $\lambda_k = \sum_{g=1}^{K} \lambda_{kg}.$

Λ_T = the sum of the arrival rates to all floors; that is, $\Lambda_T = \sum_{k=1}^{K} \lambda_k.$

p_{kg} = the probability that a load picked up by the lift at floor k needs to travel to floor $g, (p_{kk} = 0)$; that is, where defined, $p_{kg} = \lambda_{kg}/\lambda_k$.

τ_{kg} = the deterministic loaded travel time from floor k to floor g.

σ_{kg} = the deterministic unloaded travel time from floor k to floor g.

The expected empty travel time to floor k given that the previous load was delivered to floor g, E_k^g, is determined by

$$E_k^g = \sum_{i=1}^{K} p_{gi}\sigma_{ik}, \tag{32}$$

and E_k, the unconditional expected empty travel time to floor k, is given by

$$E_k = \sum_{g=1}^{K} E_k^g \frac{\lambda_g}{\Lambda_T}. \tag{33}$$

Similarly, $E_k^{(2)}$, the second moment of the empty travel time to floor k is obtained as follows:

$$E_k^{(2)} = \sum_{i=1}^{K}\sum_{j=1}^{K} p_{ij}\sigma_{jk}^2 \frac{\lambda_k}{\Lambda_T}, \tag{34}$$

where σ_{jk}^2 is the squared deterministic unloaded travel time from floor j to floor k.

Let L_k and $L_k^{(2)}$ be the first and second moments, respectively, of loaded travel time for a load that is picked up at floor k. The expressions for L_k and $L_k^{(2)}$ are obtained as follows:

$$L_k = \sum_{g=1}^{K} p_{kg}\tau_{kg}, \tag{35}$$

$$L_k^{(2)} = \sum_{g=1}^{K} p_{kg}\tau_{kg}^2, \tag{36}$$

where τ_{kg}^2 is the squared deterministic loaded travel time from floor k to floor g.

Since the service time for a load picked up at floor k is the sum of empty travel time to floor k and the loaded travel time from floor k to the destination floor, we can obtain S_k and $S_k^{(2)}$, i.e., the first and second moments of the service time of a load picked up at floor k, as follows:

$$S_k = E_k + L_k, \tag{37}$$

$$S_k^{(2)} = E_k^{(2)} + L_k^{(2)} + 2E_kL_k, \tag{38}$$

where the terms of (37) and (38) were expressed previously. Therefore, the expected lift utilization is obtained as follows:

$$\rho = \sum_{k=1}^{K} \lambda_k S_k, \tag{39}$$

which corresponds to (26).

The expected time a load waits at floor k, W_k, is equal to the expected time the load waits for service, plus the expected empty travel time for the lift to actually pick up the load after service has begun. Thus, W_k is obtained as follows:

$$W_k = \frac{\sum_{g=1}^{K} \frac{\lambda_g S_g^{(2)}}{2}}{1-\rho} + E_k. \tag{40}$$

Lastly, the overall expected waiting time for all loads assigned to the lift, W, is given by

$$W = \sum_{k=1}^{K} \lambda_k W_k. \tag{41}$$

References

[1] Armour, G. E. and E. S. Buffa, "A Heuristic Algorithm and Simulation Approach to Relative Location of Facilities," *Management Science*, Vol. **9**, 1963, pp. 294-309.

[2] Agnihothri, S. R., Narasimhan, S., and H. Pirkul, "An Assignment Problem with Queueing Time Cost," *Naval Research Logistics*, **37**, 231-244 (1990).

[3] Bozer, Y. A., Meller, R. D., and S. J. Erlebacher, "An Improvement-Type Layout Algorithm for Single and Multiple Floor Facilities," Technical Report 91-11, The University of Michigan, Ann Arbor, Michigan, 1991.

[4] Cho, M. "Design and Performance Analysis of Trip-Based Material Handling Systems in Manufacturing," Ph.D. Dissertation, Department of Industrial and Operations Engineering, The University of Michigan, Ann Arbor (1990).

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

[6] Garey, M. R. and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*, W. H. Freeman and Company, New York, New York, 1979.

[7] Graves, G. W. and A. B. Whinston, "An Algorithm for the Quadratic Assignment Problem", *Management Science* **17**, 453-471 (1982).

[8] Golden, B. L. and C. C. Skiscim, "Using Simulated Annealing to Solve Routing and Location Problems," *Naval Research Logistics Quarterly*, **33**, 261- 279 (1986).

[9] International Business Machine Corporation, "Optimization Subroutine Library: Guide and Reference Manual, Release 2," Publication SC23-0519-02, 1991.

[10] Jajodia, S., Ioannis, M., Harhalakis, G. and Jean-Marie Proth, "CLASS: Computerized LAyout Solutions using Simulated annealing," *International Journal of Production Research*, **30**, 95-108 (1992).

[11] Johnson, D. S., Aragon, C. R., McGeoch, L. A. and C. Schevon, "Optimization by Simulated Annealing: An Experimental Evaluation; Part I, Graph Partitioning," *Operations Research*, **37**, 865-892 (1989).

[12] Johnson, R. V., "SPACECRAFT for Multi-Floor Layout Planning," *Management Science*, Vol. **28**, 1982, 407-417.

[13] Kaku, B. K., Thompson, G. L. and I. Baybars, "A Heuristic Method for the Multi-Story Layout Problem," *European Journal of Operational Research*, Vol. **37**, 1988, pp. 384-397.

[14] Kusiak, A. and S. S. Heragu, "The Facility Layout Problem," *European Journal of Operational Research* **29**, 229-251 (1987).

[15] Liggett, R. S. and W. J. Mitchell, "Optimal Space Planning in Practice," *Computer Aided Design*, Vol. **13**, 1981, 277-288.

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

[17] Meller, R. D. and Y. A. Bozer, "Solving the Facility Layout Problem with Simulated Annealing," Technical Report 91-20, The University of Michigan, Ann Arbor, Michigan, 1991.

[18] Montreuil, Benoit, "A Modelling Framework for Integrating Layout Design and Flow Network Design," *Proceedings from the Material Handling Research Colloquium*, Hebron, Kentucky, 43-58 (1990).

[19] Pire, V., "Automated Multistory Layout System", Masters Thesis, Industrial Engineering Department, University of Houston, Houston, Texas, December (1989).

[20] Rosenblatt, M. J. and D. H. Kropp, "The Single Period Stochastic Plant Layout Problem," *IIE Transactions*, **24**, 169-176 (1992).

[21] Seehof, J. M. and W. O. Evans, "Automated Layout Design Program," *Journal of Industrial Engineering*, Vol. **18**, 1967, pp. 690-695.

[22] Wilhelm, M. R. and T. L. Ward, "Solving Quadratic Assignment Problems by 'Simulated Annealing'," *IIE Transactions*, **19**, 107-119 (1987).