BEBR
FACULTY WORKING
PAPER NO. 90-1633

Flow Network Design for Manufacturing Systems Layout

Dilip Chhajed
Benoit Montreuil
Timothy J. Lowe

College of Commerce and Business Administration
Bureau of Economic and Business Research
University of Illinois Urbana-Champaign

FACULTY WORKING PAPER NO. 90-1633

College of Commerce and Business Administration

University of Illinois at Urbana-Champaign

February 1990

Flow Network Design for Manufacturing Systems Layout

Dilip Chhajed*

Benoit Montreuil**

Timothy J. Lowe***

*Department of Business Administration, University of Illinois at Urbana-Champaign, 1206 South Sixth Street, Champaign, IL 61820

**Department of Operations and Decision Systems, Laval University, Quebec, Canada G1K 7P4

***Department of Management Science, University of Iowa, Iowa City, IA 52242

FLOW NETWORK DESIGN FOR MANUFACTURING SYSTEMS LAYOUT

Abstract

One way to solve a facilities layout planning problem is to use a component approach; the components being a) block design, b) input/output station location, c) material flow network design, and d) aisle netting out (accounting for aisle space). Material flow network design is an important component of this approach. In this paper a shortest rectilinear flow network problem is defined and formulated as an optimization problem. A Lagrangian relaxation of the problem gives separable, linear time solvable, shortest path problems. A heuristic, with ties to this formulation, is presented. An expression for the gap between the heuristically derived solution and the optimal solution is given. Some computational experiments using the heuristic are reported.

FLOW NETWORK DESIGN FOR MANUFACTURING SYSTEMS LAYOUT

Preleminaries

Recently, renewed interest in the efficient design of facilities has developed, in part, because of increased global competition in manufacturing and an increased consciousness toward reducing manufacturing costs. By effectively arranging and coordinating physical facilities, the greatest efficiency of men, machines, and materials, supporting a business operation, can be realized. Tompkins and White (1984) suggest that about 20% to 50% of the total operating expenses within manufacturing are attributed to material handling. Efficient facilities design can reduce these costs by at least 10% to 30% (Tompkins and White, 1984) and thus increase productivity. Facilities design has three interrelated components: structure, layout, and the determination of a (network) system to support material flow interaction between facilities, e.g. material handling systems. For best results, decisions regarding these components should not be made in isolation. However, at present, a decomposition approach is usually taken, e.g. the material handling system (network) is determined after the structure (size and shape) and layout (orientation of facilities) has been determined.

Classical layout design approaches have primarily focused on laying out the block plan rather than a detailed layout. A block plan is a scaled diagrammatic representation of the building, and normally shows the location of internal partitions and columns (e.g. Figure 1) (Foulds, 1983). Activities which are to be grouped together and physically located together are called departments. The two-letter codes in the example block layout in Figure 1 correspond to departments. See Figure 5 for department names corresponding to these codes. Interactions between pairs of departments are given in terms of flows or preferences and the layout objective is to have pairs of departments which have large flows between them close to each other thus minimizing the sum (over all pairs of departments) of the flow quantities times the distance between the departments.

Two important analytical approaches to facilities layout planning have been

suggested in the literature. First is the quadratic assignment formulation (Koopmans and Beckmann ,1987; Hillier and Connors, 1966) and the second is based on graph theory (Foulds, 1983; Giffin, 1984). However it has been shown that both of these approaches are computationally intractable (Giffin, 1984; Sahni and Gonzalez, 1976). Thus, heuristic solution procedures have been developed to solve real-sized problems. Foulds (1983), and Kusiak and Heragu (1987) give a comprehensive survey of various research efforts in this direction.

The approaches discussed above derive only a block plan. Operational details such as circulation regions, aisle structure and the location of departmental input/output stations are generally not modeled. Recently, several researchers have come to recognize that considering aisle travel explicitly in the major layout design phase provides significant potential for improvement in flow travel (via aisles) and space devoted to the aisle system. O'Brien and Abdul Barr (1980) proposed a layout improvement algorithm named S-ZAKY which computes the expected flow distance savings based on the location of the input/output stations. The INTALA interactive layout software of Warnecke and Dangelmaier (1983), which uses a construction approach, permits three internal configurations (shape with input/output locations) for each department. During the interactive design process, INTALA selects the configuration most suited to the already existing material flow structure and partial layout. Instead of directly generating a layout with aisles using a myopic approach, Montreuil and Ratliff (1988(a)) and Montreuil (1987) have proposed a component approach to layout design where the components are a) block plan, b) input/output station location, c) flow network design, and d) aisles netting out (defined below).

In the above two papers, initially a block plan is generated. Given a block plan, departmental input/output station locations can then be determined (Figure 2). All material flow between departments occurs through these stations. Given the block layout and the location of input/output stations, an optimum flow network must be designed (Figure 3). This network is the basic structure on which the flow of material and people will take place. Finally, this network is transformed to a set of aisles and circulation areas (Figure 4). The circulation area is the area which accommodates the movement of people, material and

equipment through or by a department from/to another department or input/output station. The area of each department considered during the development of the block layout is the *gross* area which consists of the *net* area plus an estimate of the needed *circulation* area (10% to 25% of net area (Hicks, 1987)). The flow network is expanded (in width) to appropriately *net-out* the area from each department and at the same time keep the width of the aisles proportional to the amount of flow and the type of flow (for example, heavy parts versus human beings) using it.

The block design can be generated by using any of the classical approaches mentioned earlier (Foulds, 1983) or by using the layout methodology proposed by Montreuil and Ratliff (1988(b)). In this latter approach a cut tree (Gomory and Hu, 1961) is used as a design skeleton to support interactive layout generation. A polynomially solvable model for optimally locating the input/output stations, when all the departments are rectangular, is given in Montreuil and Ratliff (1988(a)). Montreuil and Venkatadri (1988) have developed a comprehensive linear programming model generating a net layout from an aggregate layout coupled with a flow network.

In our paper we concentrate on the flow network component of the layout design problem (c) above). The outline of the rest of the paper is as follows. In Section 1, we discuss various topologies of networks. In Section 2 we give a brief review of the literature related to flow network design as it relates to the layout design problem. In Section 3 we define the problem of Shortest R-Flow Network (SRFN) design and present a formulation of this problem in Section 4. We present a two phase heuristic for the problem of SRFN in Section 5 where theoretical justification along with some properties of the heuristic are also given. We then report some computational experience with the heuristic in Section 6, and provide some concluding remarks in Section 7. Appendix 1 contains formal statements of the algorithms, and Appendix 2 gives details on finding a shortest path in a grid graph.

1 Flow Networks

In this paper we concentrate on the flow network component of the layout design problem. Before the flow network design can be attempted, careful thought on various issues is required. For example, some situations require the bulk of interdepartmental material handling to occur along departmental boundaries (contours). On the other hand, if there is flexibility in laying out production equipment within the block design, the aisle structure may not be required to follow the departmental contours. In such a situation, the aisle structure is permitted to be of any form. We denote such situations as *free flow*. The travel norm is also of consideration during the network design phase. Let $p(s_u)$ and $q(s_u)$ represent the horizontal-coordinate and the vertical-coordinate, respectively, of an input/output station, s_u. Then $d(s_u,s_v)= abs(p(s_u) - p(s_v)) + abs(q(s_u) - q(s_v))$, where abs(.) is the absolute value function, is the length of a shortest path between stations s_u and s_v when the rectilinear norm is used; whereas $\sqrt{(p(s_u)-p(s_v))^2+(q(s_u) - q(s_v))^2}$ is the length of a shortest path when the Euclidian norm is used. The designer may want the length of the path of actual flow between s_u and s_v , on the flow network, to be within a certain percentage of the length of a shortest path between s_u and s_v (under the particular travel norm assumptions). Thus the flow network can have various topologies (Gaskins and Tanchoco, 1987; Maxwell and Muckstadt, 1982). The appropriate topology may be selected by seeking answers to questions such as the following, for the particular layout problem under consideration:

1. Flow Assumption - Is flow permitted to occur only along departmental contours, or is it permitted to flow anywhere (free flow) through the building enclosing the departments ?
2. Direction - Does the network permit only unidirectional flows, or can it support bi-directional flows ?
3. Norm - When free flow is permitted, is the travel norm Euclidian, rectilinear, or some other norm ?
4. Travel Distance - Is every flow required to take a path of the shortest length (under the travel norm assumption) or a path whose length is within a certain percentage of the length

of a shortest path, or is no restriction placed on the distance used by a flow ?

5. Redundancies - Is the designer required to include redundant links in the flow network for greater reliability during peak traffic and blockage situations ?

The general objective of the network design phase is to minimize the sum of the fixed cost of links (network construction) and the variable cost of flows.

2 Related Work

Maxwell and Muckstadt (1982) discussed the need for the design of a flow network, which they called a track layout, and presented a method to determine the number of vehicles to support the flow requirement on a given track layout. In (Maxwell and Wilson, 1981), a method to analyze a given track layout was given, which is useful to investigate the dynamic performance of any flow network, including those designed based on our models.

Gaskin and Tanchoco (1987) gave a mathematical programming formulation of flow path design for automated guided vehicle systems. They dealt with the design of directed flow paths with the objective of minimizing the total travel between departments. Their formulation involved variables (discrete variables) for arcs only. Unlike their formulation, we will have continuous flow variables in addition to discrete arc variables. In (Gaskin and Tanchoco, 1987) no specific solution technique to obtain a solution (except enumeration or branch and bound) was given.

Egbelu and Tanchoco (1986) discussed the merits of deploying bi-directional flow paths versus uni-directional flow paths. Their simulation experiments suggest significant savings in the number of vehicles required and in the production time, at the cost of increased control requirements. Gaskin, et al. (1989) developed an integer programming formulation for the problem of optimal flow path design for an automated guided vehicles system with a virtual flow path and multiple lanes of vehicle flow.

Ho, Vijayan, and Wong (1990) considered the problem of constructing a minimal rectilinear Steiner tree of a given set of points, starting from a minimum spanning tree. The minimum spanning tree specifies the pairs of points which must be connected by a rectilinear path and hence plays the same role as the flow set (which specifies which points have material flow between them) during the flow network design stage of plant layout. Ho, Vijayan, and Wong gave a polynomial time solution procedure to construct the rectilinear Steiner tree, which is a special case of the flow network we are about to discuss.

We now define a new problem in flow network design and present a solution technique for it.

3 Shortest R-Flow Network

We consider one particular design topology which is extremely useful during the design phase of most layouts. This topology assumes free flow, an undirected network, and the rectilinear travel norm. A path of length $abs(p(s_u) - p(s_v)) + abs(q(s_u) - q(s_v))$ between stations s_u and s_v consisting of arcs parallel to the axes is called an *r-path*. We require that for every flow there exists an r-path on the network and no redundant links are required in the network (we note that the designer may add some redundant links after the optimal design is found, based on his own post-design analysis). Figure 3 shows such a network for the flows in Figure 5 and station locations in Figure 2.

The objective is to minimize the cost of a network (fixed cost) which will permit flow between all pairs of stations which have flows between them, subject to the above assumptions. Since in such a network every flow takes an r-path, the variable cost of day-to-day material handling is minimum and thus we focus on minimizing the fixed cost. In this paper we take length of the network as a surrogate for cost and minimize the total length of the r-flow network. We call the the underlying design problem the *shortest r-flow network problem* (*SRFNP*) and the optimal network is called the *shortest r-flow network* or *SRFN*.

4 Formulation of SRFNP

We first reiterate the statement of the shortest r-flow network problem. Given a set of stations S (Input/Output (I/O) points of departments); their locations on a two dimensional coordinate system, $(p(s_u), q(s_u))\ \forall\ s_u \in S$; and a set of pairs of stations $F = \{f=(s_u, s_v) :$ there is a flow between s_u and $s_v\}$; we desire to construct a network on which each flow is connected via an r-path and the network is of minimum length. We will use subscripts u and v to refer to departments. In this paper we concentrate on minimizing the total length of the network (sum of lengths of all arcs in the network), but our approach is extendable to the case when each link may have a cost not necessarily proportional to its length. Note that once the locations of the stations are given, the contours of the departments can be ignored in the SRFN problem because of the free flow assumption. Hence, throughout our discussion, department contours will not be of concern to us. Also, we will occasionally refer to a flow by the symbol f, with the understanding that the flow is defined by a pair of stations.

4.1 Initialization

Given the station locations $(p(s_u), q(s_u))$ for each $s_u \in S$ on a two dimensional coordinate system, draw a horizontal and a vertical line through each station (Figures 6, 7). Number the vertical lines 1, 2, ... from left to right and the horizontal lines 1, 2, ... from bottom to top. The intersection of each horizontal line and vertical line defines a grid point. The grid points, the station set and the collection of horizontal and vertical lines define a grid graph, G (see Figure 7). The node set V(G) of this grid graph consists of the station set and the grid points. Two nodes are connected by an arc in the arc set E(G) of this grid graph if and only if the nodes lie on the same horizontal (vertical) line and consecutive vertical (horizontal) lines. Grid graph G is said to be defined by stations in the set S. Initially, it is convenient to think of all the arcs in E(G) as being painted yellow. We will eventually change the color of some of these arcs. Let vl(*) and hl(*) refer to the vertical and the horizontal line passing through node *, respectively. For any arc a, C_a represents

the length of the arc.

In Chhajed (1989) it is shown that there exists at least one SRFN contained in the grid graph defined by the set of stations, S. This enables us to reduce the optimization problem on the Cartesian plane to a problem on the grid graph, G. It can also be shown that some of the arcs in G can be removed without affecting the value of the solution to SRFN. For example, in Figure 11, any flow using the shaded arcs can use the darkened arcs. Thus the shaded arcs can be deleted. Such reduction can be systematically done by finding the Rectilinear Hull of the stations and taking its intersection with G. The Rectilinear Hull is defined and an O(n*log(n)) procedure for computing it is given in Chhajed and Chandru (1988). This reduces the graph in Figure 7 to the one given in Figure 9 (This Figure also shows modified costs on the arcs which are computed by a method to be discussed later). From now on, G will refer to this reduced grid graph.

4.2 Preprocessing

Before we solve the problem using an optimization model or a heuristic, we preprocess the data to reduce the size of the problem. The following notations are needed for describing the preprocessing step. For each flow f, let R(f,G) represent the sub-graph formed by intersection of G with the smallest rectangle (having sides parallel to the axes) enclosing f. Thus R(f,G) consists of precisely those arcs of E(G) which may be used by f when some r-path is taken. For example, if G is the graph in Figure 9 and f =(S3, S6) then R(f,G) is given in Figure 10. For f =(S1, S3), Figure 11 shows R(f,G). As Figure 11 shows, R(f,G) may not be "rectangular" for every flow f. Given a set of weights for every arc in G, the *cost* of a path is the sum of the weights of the arcs in the path. Let S(f,G,$\underline{c}$) be a *least-cost r-path* for flow f with arc weights given by vector $\underline{c}$. Henceforth we will write a least-cost r-path as lcr-path. An algorithm called LEASTCOST to find an lcr-path is given in Appendix 2. In this algorithm we assume that R(f,G) is "rectangular" as shown in Figure A1 in Appendix 2. If some of the edges are missing, we add additional arcs to complete the rectangle and assign an arbitrary large weight M to each of these added arcs in order to apply the algorithm. The set of arcs used by a path are called *path-arcs*. The path-arcs of an lcr-path correspond to an r-path of f in G. When each arc has unit weight, every

r-path is also an lcr-path, and each such path is denoted by S(f,G,1). We now give a verbal description of the preprocessing algorithm. The reader is referred to Algorithm PREPROCESS (see Appendix 1) for an exact description of the algorithm.

We examine each flow $(s_u,s_v) \in F$ to see if both stations lie on the same horizontal or vertical line. Clearly, for such a case, the r-path for the pair is unique and it is the straight line segment, joining the two stations. Thus all the arcs on this path will be in any optimal solution. We repaint these arcs red and set the cost of these arcs to zero. This process is called *fixing* arcs. We also delete the flow (s_u,s_v) from the flow set F. For example, in Figure 9 the arcs on the unique path between (S4, S5) can be fixed and this is shown by dark lines.

We note that a unique r-path between s_u and s_v for $(s_u,s_v) \in F$ may exist in R(f,G) even when the two stations are not on the same line. This occurs if there is a single r-path between the stations in the rectilinear hull (see Section 4.1). In this case also, we fix (paint red) all the path-arcs and delete the flow from F.

At the end of this process if there exists an r-path in G for a remaining flow-pair in F using only red arcs, then the flow is removed from F. If F becomes empty then the set of red arcs is an optimal solution to our problem and we can stop. If not, we proceed to the next step.

4.3 Formulation

We now formulate the SRFN problem as a mixed integer programming optimization program (Magnanti and Wong, 1984). For each flow $f=(s_u, s_v) \in F$, if $p(s_v) < p(s_u)$ then replace (s_u, s_v) in F by (s_v, s_u). Thus, the first entry s_k of a pair $(s_k,s_l) \in F$ satisfies $p(s_k) < p(s_l)$. Partition the flow set into two sets, F^+ and F^-, as follows: If we draw a line segment joining stations s_u and s_v of flow (s_u, s_v) and the slope of this line is strictly positive then $(s_u, s_v) \in F^+$, otherwise $(s_u, s_v) \in F^-$. In other words,

$$F^+ = \{(s_u, s_v) \in F: \frac{q(s_v) - q(s_u)}{p(s_v) - p(s_u)} > 0\} \text{ and}$$

$$F^- = \{(s_u, s_v) \in F: (s_u, s_v) \notin F^+\}.$$

Although our problem calls for design of an undirected network, we form a separate directed graph for each $f \in F$. This is done as follows: For each $f \in F^+$ we create a copy of R(f,G), call it G^f, and direct the arcs in G^f toward the East and North, i.e., all vertical arcs are directed Northward and all horizontal arcs are directed Eastward. Similarly, for each $f \in F^-$ we create a copy of R(f,G), call it G^f, and direct the arcs in G^f toward the West and North (Figure 12). Let $E(G^f)$ and $V(G^f)$ denote the arcs and nodes of G^f. For each f we designate the station with lower vertical-coordinate as the *source* and the station with higher vertical-coordinate as the *sink*.

Let M^f be the node-arc incidence matrix of the directed graph G^f we have just defined for flow f. Thus M^f has $|E(G^f)|$ columns and $|V(G^f)|$ rows. An entry μ_{ij} of M^f corresponding to i^{th} row (node) and j^{th} column (arc) has a value of +1 if the arc corresponding to the j^{th} row is incident on the node corresponding to the i^{th} column, -1 if the arc corresponding to the j^{th} row is originating from the node corresponding to the i^{th} column, and zero otherwise. Let b^f correspond to a column vector having a -1 in the position of the source node, a +1 in the position of the sink node and a 0 in all other positions corresponding to other nodes of G^f. For any arc a in G, the corresponding arc in the f induced network G^f, if it exists, is denoted by a_f. We will refer to a as the *original arc* and a_f as the *induced arc* corresponding to flow f and arc a. When no confusion exists we will use a to refer to an induced arc as well. For example, $a \in E(G^f)$ refers to the induced arc corresponding to flow f and original arc a. The induced arcs are directed, the direction of which is uniquely determined by the type of flow (membership in F^+ or F^-) and the type of arc (horizontal or vertical). Two induced arcs a_f and $a_{f'}$ corresponding to flows f and f' may have different directions.

In our formulation we have two kinds of variables. For each arc $a_f \in E(G^f)$, we have continuous flow variables y_a^f which represent the amount of flow f on induced arc a_f. For each arc $a \in E(G)$ we have a 0-1 variable x_a, where $x_a = 1$ if arc a is chosen, 0 otherwise. y^f is a column vector of variables y_a^f with entries corresponding to $a_f \in E(G^f)$. Finally, for each arc a, we set C_a to the length of arc a if arc a is not fixed, and 0 otherwise. We now have the following formulation:

$$(P) \quad \min \quad \sum_{a \in E(G)} C_a x_a$$

Subject to:

$$M^f y^f = b^f \qquad \forall\, f \in F \qquad (1)$$

$$y_a^f \leq x_a \qquad \forall\, f \in F,\ a_f \in E(G^f) \qquad (2)$$

$$x_a = \{0,1\} \qquad (3)$$

$$y_a^f \geq 0 \qquad (4)$$

Here constraints (1) are the flow balance equations. Constraints (2) permit flow on an arc if and only if the arc is selected. Constraints (3) force an arc either to be in the solution or out of the solution and (4) are the non-negativity constraints on the flow variables.

The number of constraints in the above formulation can be reduced by generating an aggregate formulation, where constraints of type (2) are aggregated for all flows that could potentially use an arc. This gives:

$$(P') \quad \min \quad \sum_{a \in E(G)} C_a x_a$$

Subject to:

$$M^f y^f = b^f \qquad \forall\, f \in F \qquad (5)$$

$$\sum_{f \in F_a} y_a^f \leq |F_a| x_a \qquad \forall\, a \in E(G) \qquad (6)$$

$$x_a = \{0,1\} \qquad (7)$$

$$y_a^f \geq 0 \qquad (8)$$

where F_a is the set of flows which can use arc a.

Problems P and P' are equivalent in the sense that the optimal objective functions of both have the same value. Let (P_{LP}) and (P'_{LP}) refer to the LP relaxation of problems (P) and (P'), respectively. It can be shown that $Z^*(P_{LP}) \geq Z^*(P'_{LP})$, where $Z^*(P_{LP})$ is the optimal solution value of (P_{LP}) and $Z^*(P'_{LP})$ is the optimal solution value of (P'_{LP}). Thus, although (P) has more constraints, it is preferred over (P') (Magnanti and Wong, 1984; Dror, et al., 1988) because while solving the original problem by branch and bound, lower bounds are frequently generated by LP relaxation, in which case one prefers the formulation which generates better bounds.

4.4 Lagrangian Relaxation

We now explore the possibility of solving (P) by forming its Lagrangian dual by

relaxing constraints (2) and including them in the objective function with multipliers v_a^f. We denote the vector of these multipliers as $\underline{v}$. This gives the following relaxed problem:

$$(P(\underline{v}))\quad \min \sum_{a\in E(G)} (C_a - \sum_{f\in F_a} v_a^f)\, x_a + \sum_{a\in E(G)} \sum_{f\in F_a} v_a^f\, y_a^f$$

Subject to: (1), (3) and (4).

The ideal choice of multipliers is such that they solve the Lagrangian dual problem:

$$(DP(\underline{v}))\qquad \max_{\underline{v}}\ (P(\underline{v})).$$

The objective function in $P(\underline{v})$ has terms involving x_a, but there are no constraints involving x_a except for (3). We can set the value of x_a as 0 or 1 based on the sign of $(C_a - \sum_{f\in F_a} v_a^f)$. The remainder of the problem decomposes into one independent problem $(P(\underline{v})_f)$ for each flow f and is given by:

$$(P(\underline{v})_f)\quad \min \sum_{a\in E(G^f)} v_a^f\, y_a^f$$

Subject to: $M^f y^f = b^f$

$y_a^f \geq 0$

We note that each of these problems is equivalent to a shortest path problem. Because the underlying network is a grid network, we can solve each of these shortest path problems in O(m) time where m is the number of nodes in R(f,G) (see Appendix 2). Finally, we want to select the multipliers so that $(P(\underline{v}))$ is maximized, which will provide a lower bound to (P). The dual problem can be solved by using, for example, the subgradient approach (Fisher, 1981). We note that the optimal value of the x_a variables in $(P(\underline{v}))$ is unaffected when constraints (3) are relaxed and each x_a is constrained to be between 0 and 1. Thus the Lagrangian dual problem has the *integrality property* (Geoffrion, 1974) implying that the solution of the dual will be no better than the linear programming relaxation of (P). However, this does not render the relaxation useless since the Lagrangian dual may be easier to solve than solving the LP due to the fact that each subproblem of $(P(\underline{v}))$, for fixed set of multipliers, can be solved very efficiently.

One can also find an upper bound on problem (P) by constructing a primal feasible solution while solving each problem $P(\underline{v})_f$. While finding the shortest path in $P(\underline{v})_f$ we collect the original arcs corresponding to the path-arcs of the shortest paths for each f in F. The collection of these arcs is a feasible solution. However, in this case, as the lower bound increases (via multiplier change in the dual problem), the upper bound may not be non-increasing. A remedy for this is to retain the best primal feasible solution found thus far.

In the next Section we give a simple heuristic to find a good primal solution to the SRFN problem. Unlike the above method, the heuristic is simple to implement, has ties with the above formulation, and is attractive from a computational point of view as well.

5 Heuristic

Our problem is a special case of the fixed charge network design problem which is known to be NP-complete (Johnson, et al., 1978). Although we have not proven that a fixed charge network design problem is NP-complete on a grid graph, most problems that are hard on a general graph are found to be hard on a grid graph as well (Gary and Johnson, 1977; Itai, et al., 1982). Additional evidence of the complexity of this problem comes from the fact that if we relax the requirement that each flow must take an r-path and instead allow each flow to take a path consisting of a sequence of horizontal and vertical moves, the problem of finding a minimum length network reduces to the NP-complete rectilinear Steiner tree problem (Gary and Johnson, 1977).

In this Section we present a two phase heuristic. Formal statements of the algorithms appear in Appendix 1. Subsequently, we present a theoretical justification of the heuristic. We also give cases where the Phase I solution is an optimal solution, providing further evidence of the appropriateness of the heuristic.

5.1 Phase I - Build Phase

We construct a feasible solution during the Build phase of the heuristic. As alluded to in Section 4, because of our r-path assumption, each flow defines a rectangular subgraph R(f,G) consisting of all the arcs the flow can possibly use. The nodes corresponding to the two stations (the origin and destination of f) lie on the opposite corners of this rectangular subgraph. Thus the number of flows which can potentially use an arc a can be easily determined. This number is called the *potential* of arc a and is denoted by p_a. Thus p_a is the number of rectangular subgraphs R(f,G) which contain arc a. For the red arcs, we do not calculate the potentials (these arcs are fixed in the solution) and the modified cost of each red arc will be set to zero. For each yellow arc, define its *modified cost* c_a to be the length of the arc divided by its potential (Figure 9) (note that C_a is the length of the arc and c_a, as defined above, is the modified cost). This is motivated by the fact that the larger the number of potential users of an arc, the less must be its price per user, whereas the longer the

length (or larger the cost) of an arc, the larger must be the price since our objective is to minimize the total length of the network. The modified cost of each red arc is set to zero, since these arcs are already fixed and other flows may use them at zero cost.

Next, find an lcr-path, denoted by $S(f,G,\underline{c})$, for each flow f using the modified costs $\underline{c} = \{c_a : a \in E(G)\}$ as weights. The collection of original arcs corresponding to the lcr-paths of each flow in F, along with the red arcs, is denoted by graph N (Figure 13). No additional arc is painted red during the Build phase. It is easy to see that N is a feasible solution of SRFN. We note that the lcr-path for each flow has been found in an independent fashion. That is to say that while finding the lcr-path for a given flow, the information about the original arcs collected so far (from flows considered earlier in the procedure) is ignored. One could use a variation of the algorithm in which, after finding an lcr-path for an arbitrary flow $f \in F$, the original arcs corresponding to this path are fixed. Then flow f is deleted from F and the potential and modified cost of each remaining yellow arc is adjusted to reflect this change. The procedure repeats until the set F is empty. This modification, however, increases the computational time quite substantially. Our present approach is justified in light of Phase II, where we identify and delete the redundant arcs.

If there exist multiple lcr-paths for any flow, we include in N the original arcs which correspond to the path-induced arcs by all these paths. The rationale is that redundant arcs will be deleted during the improvement phase and since it is not known which of these arcs can be used by other flows in the final solution, we include all of them at this stage. The output of Phase I is a feasible grid network N (with each arc painted either red or yellow). From now on, we only work with network N. Algorithm BUILD in Appendix 1 gives the summary of what we have just described. We shall see later (Corollary 2) under which circumstances this solution can be an optimal one. If the condition of Corollary 2 is not true, then we implement the Improvement phase.

5.2 Phase II - Improvement Phase

We now attempt to improve (in terms of the sum of the lengths of the arcs) the feasible solution generated during the Build phase. The Improvement phase has two stages: Path Improvement and Arc Improvement .

Path Improvement Stage: During this stage, we fix some additional arcs and delete those flows for which there exists an r-path consisting entirely of fixed arcs.

The approach in Path Improvement is to look at an entire path for a flow. We select an unexamined flow, f, and find all of the yellow (unfixed) arcs in N which are essential to connect stations (s_u,s_v) of f by an r-path (Figure 14). These arcs are cut arcs, denoted by CA(f,N), the removal of any one of which will disconnect all r-paths between s_u and s_v in N. In Figure 14 the arcs defined by the node pairs (f,g),(g,h), and (h,k) are cut arcs for f=(A,B). Let R(f,N) =N$\cap$R(f,G). To determine if a yellow arc a in R(f,N) is a cut arc for flow f, we check whether an r-path exists in N\a for flow f. If the answer is "No" then arc a is a cut arc. This process can be repeated for every yellow arc in R(f,N) to obtain CA(f,N).

To check whether an r-path exists for f in N\a we assign a weight of 1 to every arc in R(f,N\a), introduce (temporarily) additional arcs so that R(f,N\a) is "rectangular" and assign a very large weight M to these additional arcs. Then we use Algorithm LEASTCOST (see Appendix 2) to find an lcr-path. If the cost of this path is more than M then no r-path exists in N\a for f.

We fix the arcs in CA(f,N) (paint them red and set their cost to zero). None of the arcs will be painted red if and only if there exist two or more edge disjoint r-paths for flow f. Next we check whether there exists an r-path for f consisting of only red arcs. If such an r-path exists, we remove f from F. Otherwise we choose another unexamined flow and repeat the above until all the flows are examined.

At the end of this procedure, some of the arcs may have been fixed and the flow set may have been reduced.

Arc Improvement Stage: During the arc improvement stage, for each yellow arc a in N we determine whether a feasible solution to the SRFN problem exists in N\a. This is done by checking for each of those flows which can potentially use the arc, whether an alternate r-path exists which does not use arc a. The verbal details of the algorithm now follows.

We first recompute the potential of each yellow arc in N. However, now flow f may not contribute to the potential of every yellow arc in R(f,N). There may be some arcs

in R(f,N) which are not in any r-path for f in R(f,N). For each such arc, flow f contributes nothing to the arc's potential. For example, in Figure 14 the darkened arcs correspond to arcs in N and note that arcs defined by (d,e) and (h,i) will not be used in any r-path between A and B. Another reason for recomputing the potentials is that the flow set may be smaller now because some flows may have been deleted during the Path Improvement stage.

After recomputing the potentials, we choose a yellow arc a, with minimum potential and check whether N\a remains feasible. If this is true, i.e. N\a is feasible, then arc a can be deleted and we set N ← N\a and recompute the potentials. On the other hand, if N\a is not feasible then arc a must be present for N to remain feasible and we fix arc a (paint it red). This process is repeated using the yellow arc (from the remaining yellow arcs) having the smallest potential until no yellow arcs remain. At the end of the process all the arcs in N are fixed, implying that none of them are redundant. Figure 15 shows the result of applying the Improvement phase on the graph of Figure 13.

5.3 Justification of the Heuristic

We now relate our heuristic with the formulation and the relaxation given in Section 4. First we prove that it is optimal to set the multipliers in (DP($\underline{v}$)) such that the coefficient of x_a is zero in (P($\underline{v}$)).

Theorem 1: There exists an optimal dual solution in which $C_a = \sum_{f \in F_a} v^f_a, \forall\ a \in E(G)$.

Proof: We assume that the above condition is not true for some optimal solution v^{*f}_a and derive another solution of greater or equal value satisfying the condition. Choose an arc a in the optimal solution violating the above condition. There are two cases,

Case I: $(C_a - \sum_{f \in F_a} v^{*f}_a) = d > 0.$

In the optimal solution to (P($\underline{v}$)), x^*_a is set to 0. We increase each v^{*f}_a by an amount Δ where $\Delta = d/|F_a|$. With the new multipliers, it is still optimal to set x_a to 0. However, an increase in the cost of arcs a, v^f_a, for all the flows which can potentially use it does not decrease the shortest (cheapest) path for these flows. Hence this new solution does not decrease the solution value of the dual and makes the condition of the Theorem true for one

more arc.

Case II: $(C_a - \sum_{f \in F_a} v^{*f}_a) = d < 0$.

In this case x^*_a is set to 1 and contributes the negative term d to the value of $(P(\underline{v}))$. If we now decrease the value of each multiplier by Δ where $\Delta = -d/|F_a|$ there will be an increase in the value of $(P(\underline{v}))$ by -d since it still remains optimal to set $x^*_a = 1$. Let F°_a be the flows which use arc a with the original multipliers. Clearly with the new multipliers, each $f \in F^\circ_a$ will also use arc a and the change in the length of the shortest path will be $-\Delta$. Let F'_a be the set of flows using arc a with the new multipliers, but not using arc a with the original multipliers. The change in the value of the shortest path lengths for each $f \in F'_a$ will be negative, but no smaller than $-\Delta$. Thus the total change in shortest path lengths, over all flows, will be no smaller than $-\Delta(|F^\circ_a|+|F'_a|)$. However, $|F^\circ_a|+|F'_a| \leq |F_a|$, so that the total change is bounded below by $-\Delta(|F_a|) = d$. Thus the net change in the objective function is at least as large as (-d+d) = 0, and so changing the multipliers on arc a cannot decrease the value of the dual problem.«»

By the above theorem it is optimal to set the multipliers so that,

$$C_a = \sum_{f \in F_a} v^f_a \qquad \forall\, a \in E(G). \tag{9}$$

In addition, if we set $v^f_a = v^{f'}_a \;\; \forall\, f, f' \in F_a$, then to satisfy (9) we must have,

$$v^f_a = \frac{C_a}{|F_a|} \qquad \forall\, f \in F_a,\, a \in E(G). \tag{10}$$

But $|F_a|$ is merely the maximum number of flows which can use arc a, i.e., the potential p_a, as defined in the heuristic. In this case, the multiplier values in (10) are exactly the weights used in the Build phase of the heuristic. The heuristic subsequently finds an lcr-path for each flow, and a similar computation is also done to solve $(P(\underline{v}))$, for any choice of the multipliers. Note that in $(P(\underline{v}))$ and $(P(\underline{v})_f)$, any (directed) path from the source to the sink is an r-path due to the directions on the arcs. Hence a shortest path in $(P(\underline{v}))$ and $(P(\underline{v})_f)$ refers to an lcr-path. Thus, the sum of the costs of lcr-paths is actually a lower bound to (P) and the heuristic constructs an upper bound by considering all the path-

induced arcs. This provides a sound basis for the weights used and the steps carried out in the Build phase of our heuristic. The Improvement phase obviously tries to construct a better solution by removing arcs wherever possible.

We now show that using the heuristic with this particular choice of weights is in fact equivalent to solving the LP relaxation of the aggregate formulation (P'). To do this we first need the following lemma:

Lemma 1: Let (P'_{LP}) denote the LP relaxation of (P'), then there exists an optimal solution (x*,y*) to (P'_{LP}), in which constraints (6) are tight, i.e.,

$$\sum_{f \in F_a} y^{*f}_a = |F_a| x_a^* \quad \forall\ a \in E(G).$$

Proof: Let (x*, y*) be an optimal solution to (P'_{LP}) and suppose there exists an arc a' such that

$$\sum_{f \in F_{a'}} y^{*f}_{a'} < |F_{a'}| x_{a'}^*. \tag{11}$$

The objective function of (P'_{LP}) has terms involving only x_a , $a \in (G)$, and the coefficient of $x_{a'}$ is non-negative ($C_{a'}$ is the length of arc a'). Hence, we can reduce the value of $x_{a'}^*$ to make (11) tight without making the objective function value worse. «»

By the virtue of this Lemma, we can substitute $x_a = \frac{1}{|F_a|} \sum_{f \in F_a} y^f_a$, $\forall\ a \in E(G)$ in the objective function of (P'_{LP}) giving:

(P'_{LP}) min $$\sum_{a \in E(G)} C_a \frac{1}{|F_a|} \sum_{f \in F_a} y^f_a = \sum_{f \in F} \sum_{a \in E(G^f)} \frac{C_a}{|F_a|} y^f_a$$

Subject to: (5), (8) and

$$0 \le \frac{1}{|F_a|} \sum_{f \in F_a} y^f_a \le 1 \tag{12}$$

Since the costs are positive and we are minimizing, no y^f_a will exceed one and so $\sum_{f \in F_a} y^f_a \le |F_a|$. Hence, (12) will be automatically satisfied and can be dropped. This enables us to decompose (P'_{LP}) into a separate shortest path problem for each flow. Again

the weights on arcs in these shortest path problems are the same weights as used in the heuristic. This gives us the following theorem:

Theorem 2: The sum of the costs of lcr-paths during the Build phase of the heuristic is equal to $Z^*(P'_{LP})$.

5.4 Bounds

Let $Z^*(P(\underline{v}))$ be the optimal value of $(P(\underline{v}))$ for a particular choice of the multipliers. Let $\tilde{G}_f$ be the original arcs which correspond to the arcs in an lcr-path for flow f, and let $\tilde{G}$ be the graph where $a \in E(\tilde{G})$ if and only if $a \in \tilde{G}_f$ for some f. Let $\tilde{F}_a$ be the set of flows using arc a in their shortest path. Let $Z^*(P)$ denote the optimal solution value to problem (P) and Z^*_p denote the sum of the lengths of the arcs in $E(\tilde{G})$. We now have the following bound on the gap between the primal and the optimal solution value (see Erlenkotter (1978) for a similar bound for the uncapacitated facility location problem).

Theorem 3:

$$Z^*_p - Z^*(P) \le \sum_{a \in E(\tilde{G})} \left(C_a - \sum_{f \in \tilde{F}_a} v_a^f \right) + \sum_{a \in E(G)} \max \{0, \sum_{f \in F_a} v_a^f - C_a\} \qquad (13).$$

Proof: In solving $(P(\underline{v}))$ for a particular choice of the multipliers, it is optimal to set $x_a = 1$ if $C_a - \sum_{f \in F_a} v_a^f \le 0$, and 0 otherwise. Also, for fixed f, the length of the lcr-path is $\sum_{a \in E(\tilde{G}_f)} v_a^f$.

This gives,

$$Z^*(P(\underline{v})) = \sum_{a \in E(G)} \min \{0, C_a - \sum_{f \in F_a} v_a^f\} + \sum_{f \in F} \left(\sum_{a \in E(\tilde{G})} v_a^f \right)$$

$$= \sum_{a \in E(G)} \min \{0, C_a - \sum_{f \in F_a} v_a^f\} + \sum_{a \in E(\tilde{G}_f)} \sum_{f \in \tilde{F}_a} v_a^f.$$

Also,

$$Z^*_p = \sum_{a \in E(\tilde{G})} C_a$$

$$= \sum_{a \in E(\tilde{G})} \left[\sum_{f \in \tilde{F}_a} v_a^f + \left(C_a - \sum_{f \in \tilde{F}_a} v_a^f \right) \right]$$

$$= \sum_{a \in E(\tilde{G})} \sum_{f \in \tilde{F}_a} v_a^f + \sum_{a \in E(\tilde{G})} \left(C_a - \sum_{f \in \tilde{F}_a} v_a^f \right).$$

$$\text{So, } Z^*_p - Z^*(P) \leq Z^*_p - Z^*(P(\underline{v}))$$

$$= \sum_{a \in E(\tilde{G})} (C_a - \sum_{f \in \tilde{F}_a} v_a^f) - \sum_{a \in E(G)} \min \{0, C_a - \sum_{f \in F_a} v_a^f\}$$

$$= \sum_{a \in E(\tilde{G})} (C_a - \sum_{f \in \tilde{F}_a} v_a^f) + \sum_{a \in E(G)} \max \{0, \sum_{f \in F_a} v_a^f - C_a\}. \text{ «»}$$

We now have the following Corollaries of Theorem 3:

Corollary 1: If $\tilde{G}$ is the graph as defined above when the multipliers are set as in the Build phase of the heuristic and if Z^*_p is the sum of the lengths of the arcs in this graph then,

$$Z^*_p - Z^*(P) \leq \sum_{a \in E(\tilde{G})} C_a \left(1 - \frac{|\tilde{F}_a|}{|F_a|}\right)$$

Proof: During Build phase, we set $v_a^f = \frac{C_a}{|F_a|}$ so that $\max \{0, \sum_{f \in F_a} v_a^f - C_a\} = 0$ for all $a \in E(G)$. Thus, (13) reduces to:

$$Z^*_p - Z^*(P) \leq \sum_{a \in E(\tilde{G})} (C_a - \sum_{f \in \tilde{F}_a} v_a^f)$$

$$= \sum_{a \in E(\tilde{G})} \left(C_a - \sum_{f \in \tilde{F}_a} \frac{C_a}{|F_a|} \right) = \sum_{a \in E(\tilde{G})} C_a \left(1 - \frac{|\tilde{F}_a|}{|F_a|}\right) \text{ «»}$$

Corollary 2: If the flow on each arc in $\tilde{G}$ as defined in Corollary 1 is equal to the potential of the arc, then $\tilde{G}$ is an optimal solution to problem (P).

Corollary 2 gives us a sufficient condition for the Phase 1 solution to be an optimal solution to problem (P).

6 Empirical Study

The heuristic was programmed in Pascal language on IBM 3081 mainframe computer. The following main steps were carried out.

Data Generator: We first generated the station locations and flows. Each station's location was generated by randomly selecting the x-coordinate and the y-coordinate. These coordinate values were restricted to be integers from 1 to 25. Care was taken to make sure that no two stations in same problem have identical x- and y-coordinates. Next, a total of θ flows were generated by randomly selecting θ pairs of distinct stations. The selection was conditioned so that no two flows had an identical pair of stations and each station was used by at least one flow. For a given number, s, of stations, the number of flows (θ) was set at three different densities: a) $\theta = s$, b) $\theta = 1.5s$, and c) $\theta = 2s$. For a choice of flows and stations, five different problems were generated. In Table 1, the notation 10.15.2 in the first column refers to the second problem with 10 stations and 15 flows. We will now refer to the columns of this table without mentioning the Table itself.

Heuristic: The data generated by Data Generator is fed to the Heuristic program. First, a Phase I solution is generated. The sum of the costs of the least cost r-paths for all flows gives a lower bound (column 3). The sum of the lengths of the arcs used in these paths provides an upper bound (column 2). This upper bound serves as an input to the Improvement phase. In the Improvement phase, we simply examined each edge of the input feasible solution and checked whether deleting it would still give a feasible solution. This phase often gave an improved feasible solution (column 4) which is the output of the heuristic. Column 5 reports the time taken by the heuristic in seconds (including the input/output time).

LP Lower Bound: As we have shown in Section 5.3, the lower bound obtained in Phase I is a solution to the Lagrangian relaxation problem with one particular set of values of the multipliers. Since the best solution of the Lagrangian dual problem is the same as the LP relaxation (see Section 4.3) we computed the LP solution for our test problems. First a code generator was created to produce the constraints and the objective function of problem

(P_{LP}) for a given set of stations and flows (data generated by Data Generator). This problem was then solved using a linear programming code. Column 6 reports the LP solution and column 7 reports the number of iterations taken by the LP code to obtain the optimal LP solution.

For four of the problems solved (those with an asterisk in column 1) the heuristic did not provide a very good feasible solution to the original problem (as measured by the comparison between the upper and lower bounds. For these problems we perturbed the data to create a new (perturbed) problem, but in such a way that the heuristically generated feasible solution to the new problem is feasible to the original problem. Thus the cost of the feasible solution to the new problem is an upper bound on the cost of the optimal solution to the original problem. An example of such a perturbation is illustrated by Figure 16. Suppose that the flow set for the original problem included {(1,2),(1,3)} (Figure 16(a)). In the perturbed problem, we replaced this flow set with {(1,2),(2,3)} (Figure 16(b)). Note that any feasible solution to the new problem must contain an r-path between 1 and 2, and an r-path between 2 and 3. But, the union of these two r-paths is an r-path between 1 and 3, and so the solution is feasible to the original problem. By making such perturbations, we significantly reduced the upper bounds on these four problems. For example, on problem 12.18.2, the upper bound was reduced from 122 (original problem) to 94 (perturbed problem). We believe that other such perturbations are possible, and we intend to explore this area in our follow-on work.

In column 8, we report the percent difference between the improved upper bound an the lower bound (LP solution). The lower bound for each problem with an asterisk was generated from the LP relaxation of the original (unperturbed) problem, and so is a valid bound to the original problem.

For many problems (in fact 31 out of 45), the percent gap between the objective value of the feasible solution given by the heuristic and the linear programming lower bound is less than or equal to 5%. The largest gap found was 12.37% while the average gap was 3.92%. As column 8 indicates, for 14 problems the heuristic gave an optimal solution. For seven of these problems, the feasible solution found by the Build phase itself was optimal. This is especially interesting because the Build phase considers only one

particular set of values of Lagrangian multipliers. The average time taken by the heuristic was 1.09 seconds with the maximum time being 3.37 seconds. On the other hand, computing the linear programming lower bound (which was found to compute the gap) took a lot more time, which is reflected by the number of iterations taken by the LP code (column 7). We do not report the exact time taken by the linear programming code because some of these were solved on a different machine (some were solved on the IBM mainframe computer using LINDO and others were solved on a Sun workstation using the CPLEX linear programming code). But to give an idea, for example, the average time taken by problems with 12 stations and 18 flows was about 115 seconds on a Sun workstation and this excludes the time taken to generate the formulation and the input/output time.

We note that the primary purpose of a solution to SRFN is to provide a starting point for the design of a material handling network. The assumptions of the model, like most mathematical models, do not capture all aspects of a real design. Thus, these solutions will be "moulded" by the designer to incorporate the assumptions and conditions not explicitly considered in the model. In practice, many different block layouts are generated; for each block layout, a flow network can be developed for a given design of station locations and an "optimal" layout is chosen with respect to many tangible and intangible factors. Thus, the network design model will be used a large number of times before arriving at a final detailed layout design. Thus the model should be able to generate good solutions without consuming a large amount of time. Our heuristic is very fast, easy to program, and provides, on an average, good solutions.

7 Concluding Remarks

In this paper, we have defined the shortest r-flow network problem which has applications in the material handling network design phase of the facility layout problem. We have given a multicommodity-flow-based formulation of this problem, the Lagrangian relaxation of which gives shortest path sub-problems. We have shown that each of these shortest path problems can be solved in linear time.

We have also presented a two-phase heuristic and shown its ties with the formulation. An expression for the gap between the heuristically derived feasible solution and the optimal solution is given. One can solve a series of problems using the subgradient approach to simultaneously generate feasible solutions and lower bounds. The gap, discussed above, can give a good stopping criterion. Note that one can improve the feasible solution formed with path-induced arcs by applying the Improvement phase of the heuristic. A limited computational study showed very promising results on the quality of the solution obtained via the heuristic.

Our formulation and the heuristic is extendable to some of the other topologies discussed in Section 1. The topologies which easily fit our framework are those in which,

a) direction of travel is important,

b) some additional arcs (for example, diagonal arcs) in the grid graph are present and the flows are permitted to use them,

c) certain arcs are not available for travel for certain flows (when complete free flow is not possible due to the presence of immovable machines or other structures),

d) every flow can take a path whose length is within a certain percentage of the length of a shortest rectilinear path.

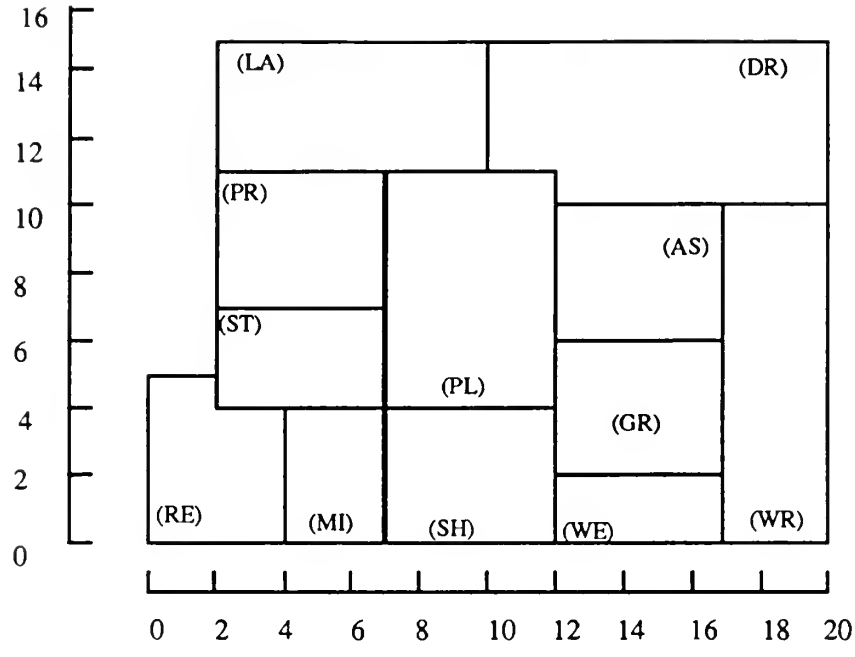


Figure 1. A Block Layout

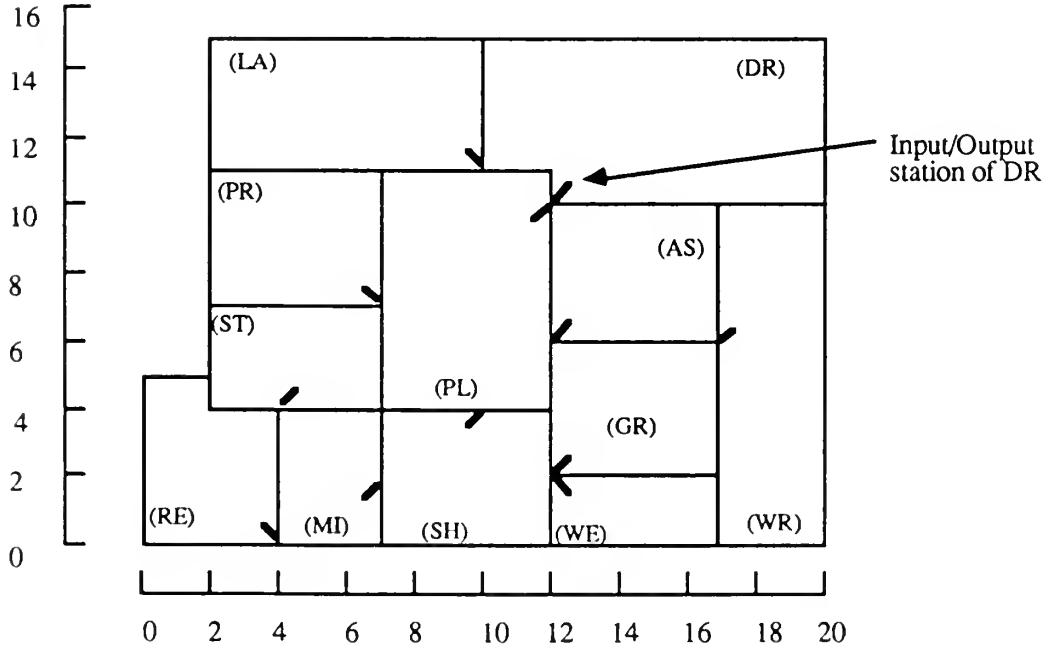


Figure 2. A Block Layout with Input/Output Stations

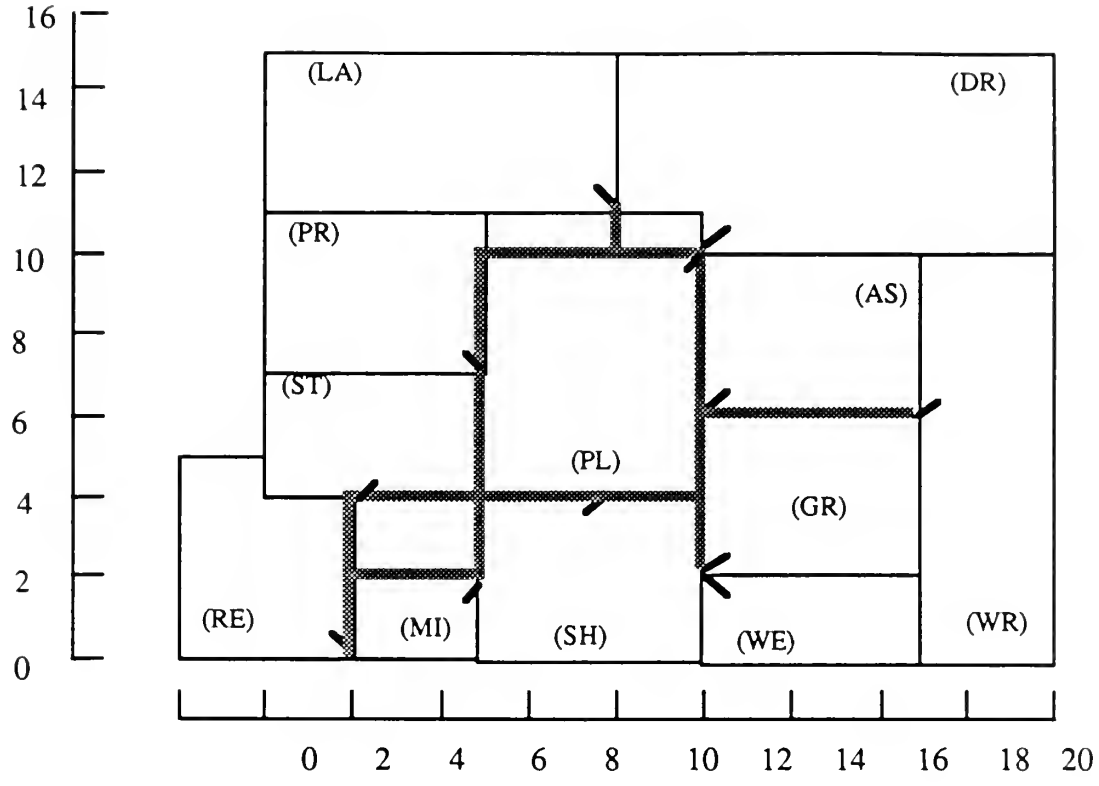


Figure 3. A Block Layout with Input/Output Stations and Flow Network

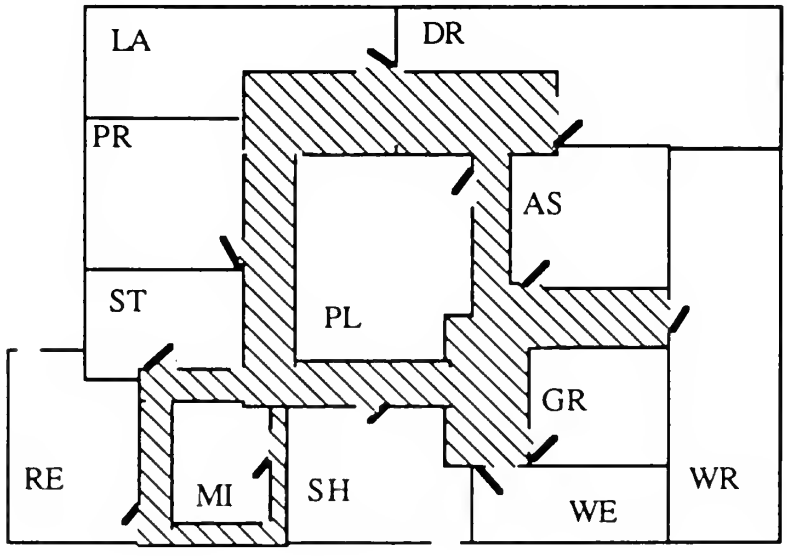


Figure 4. A Block Layout with Input/Output Stations and Aisles

DEPARTMENT	CODE	AREA	FLOW-BETWEEN RE	MI	PR	LA	DR	WE	PL	GR	AS	WR	SH	ST
RECEIVING	RE	18		30										30
MILLING	MI	12			40	10			10					
PRESSES	PR	20				35	5							20
LATHES	LA	32					20	20	45					30
DRILLS	DR	48							5	20	10			
WELDING	WE	10							5	20	15	10		
PLATING	PL	35								45	10	20		
GRINDING	GR	20									25		60	
ASSEMBLY	AS	20										50		
WAREHOUSE	WR	27												
SHIPPING	SH	20												
STORES	ST	15												

Figure 5. A Flow Matrix

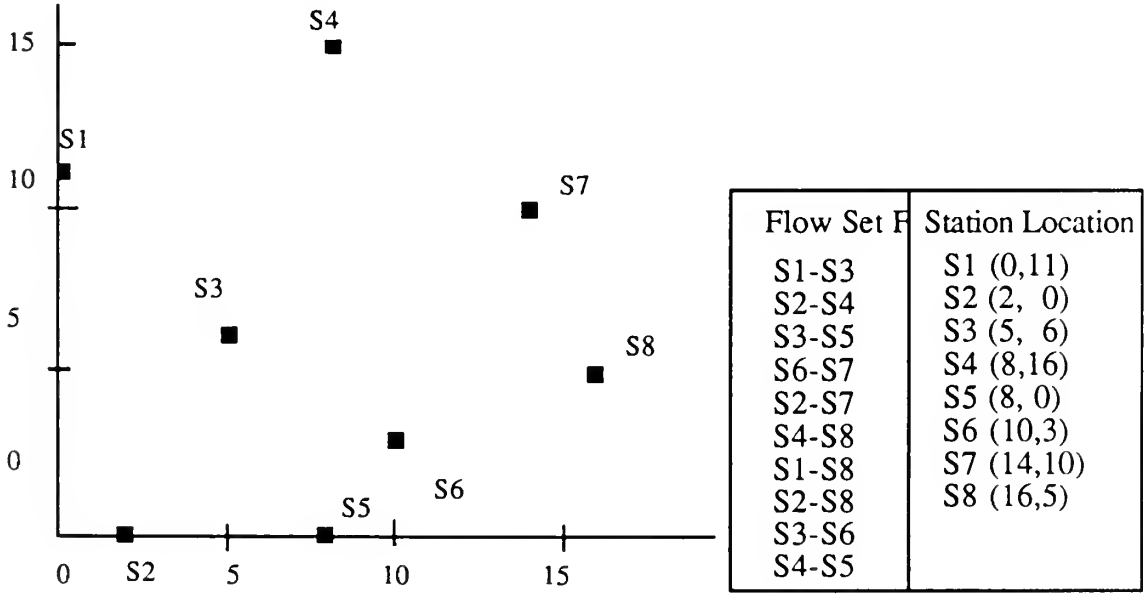


Figure 6. Flow Set, Station Set, and Station Locations

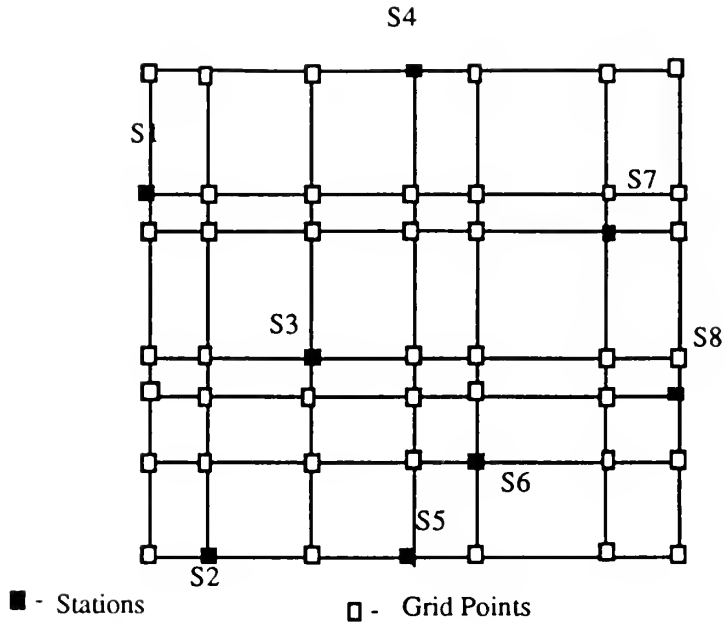


Figure 7. The Horizontal and Vertical Lines at the Station Locations Giving Grid Graph G

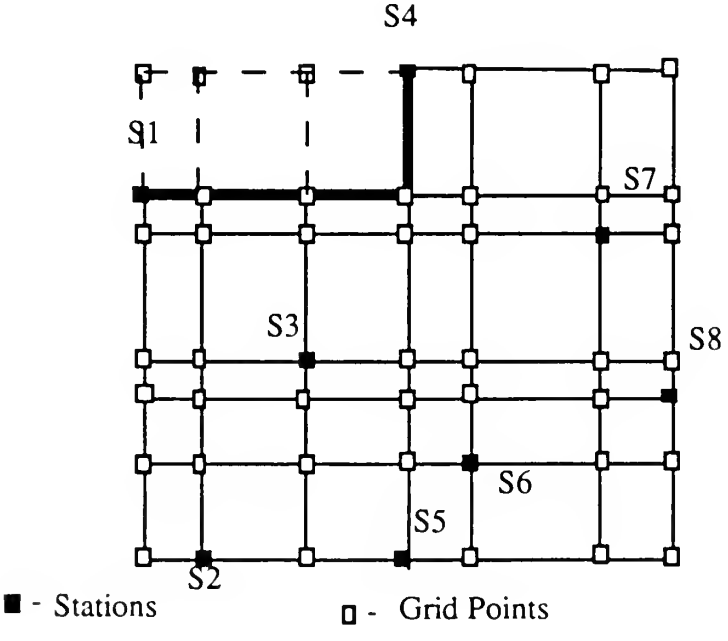


Figure 8. Figure to Illustrate Some Redundant Arcs

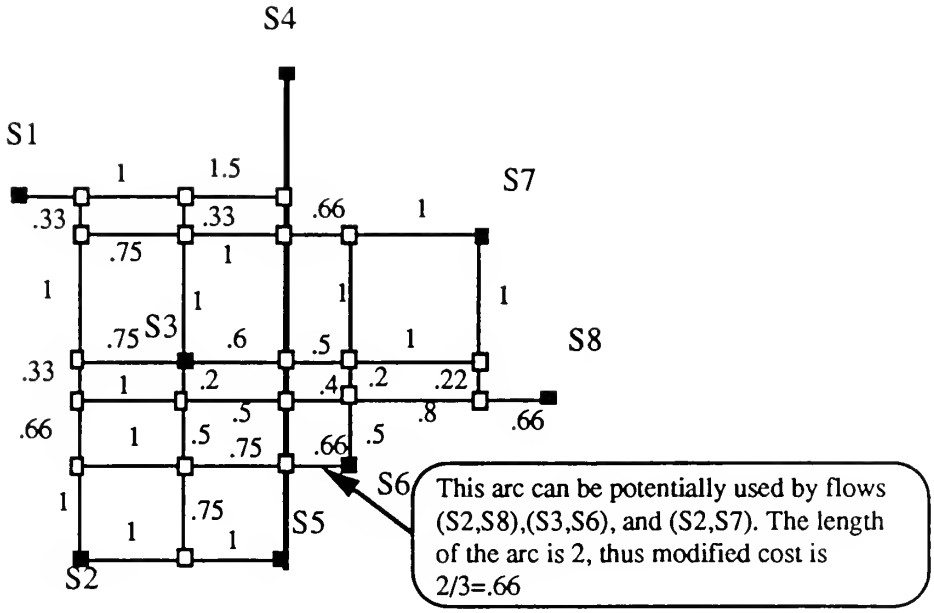


Figure 9. Reduced Grid Graph with Modified Costs c_a

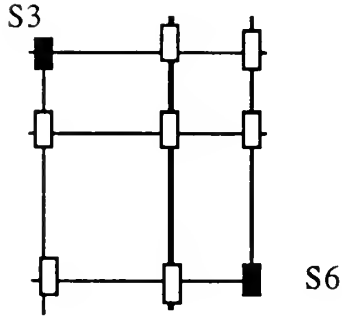


Figure 10. R((S3,S6),G)

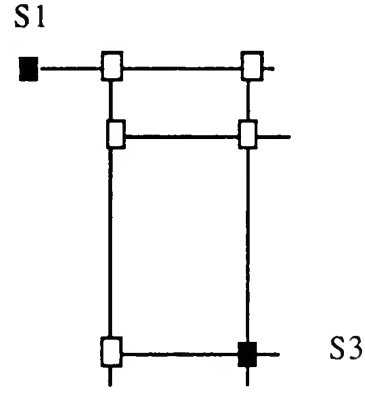


Figure 11. R((S1,S3),G)

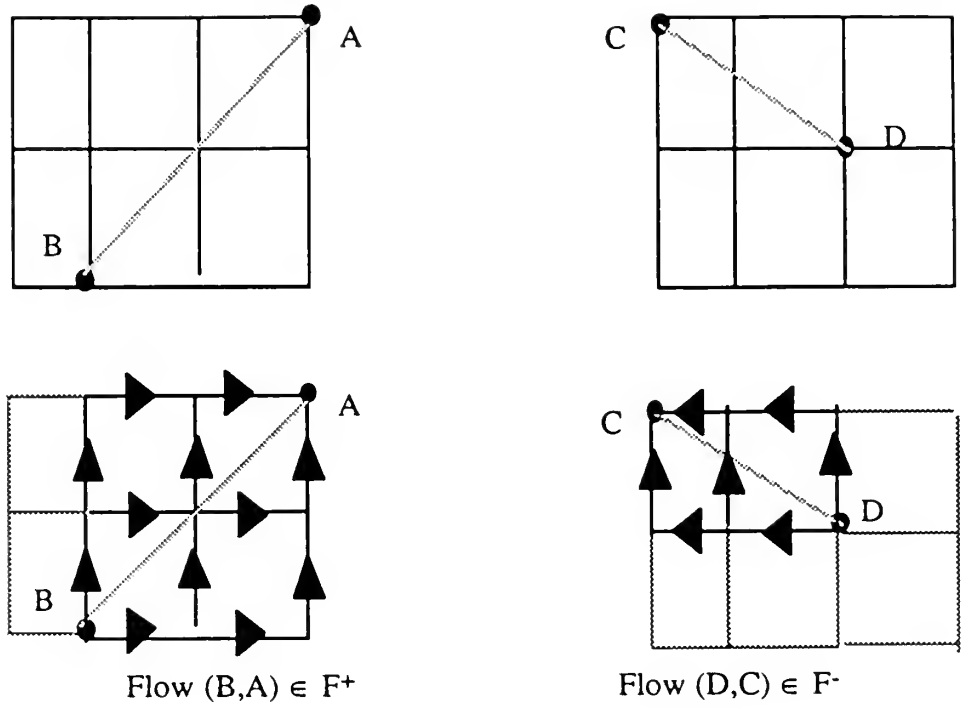


Figure 12. Formation of Network G^f

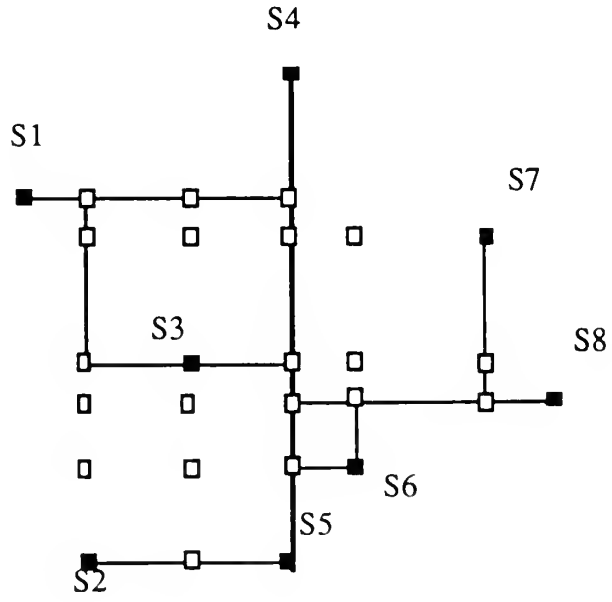


Figure 13. Feasible Solution Given by the Build Phase

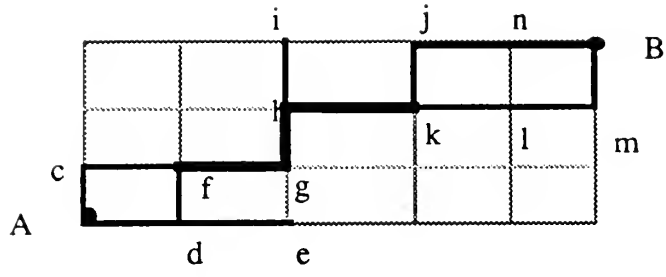


Figure 14. Figure to Illustrate Cut Arcs (cut arcs are thicker)

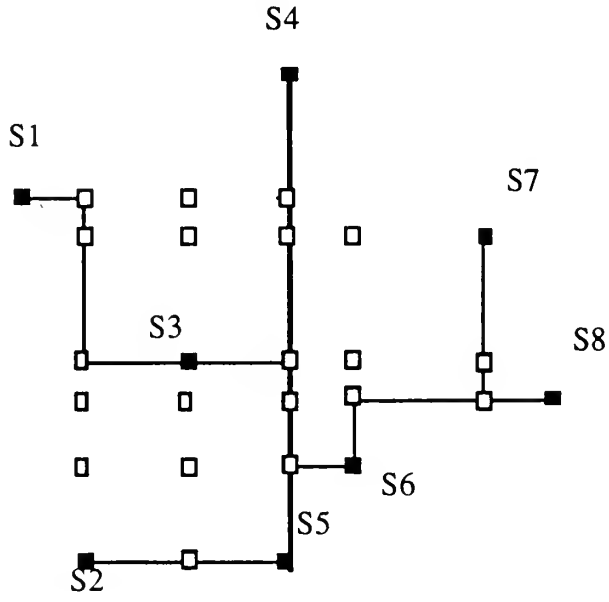


Figure 15. Output of the Improvement Phase

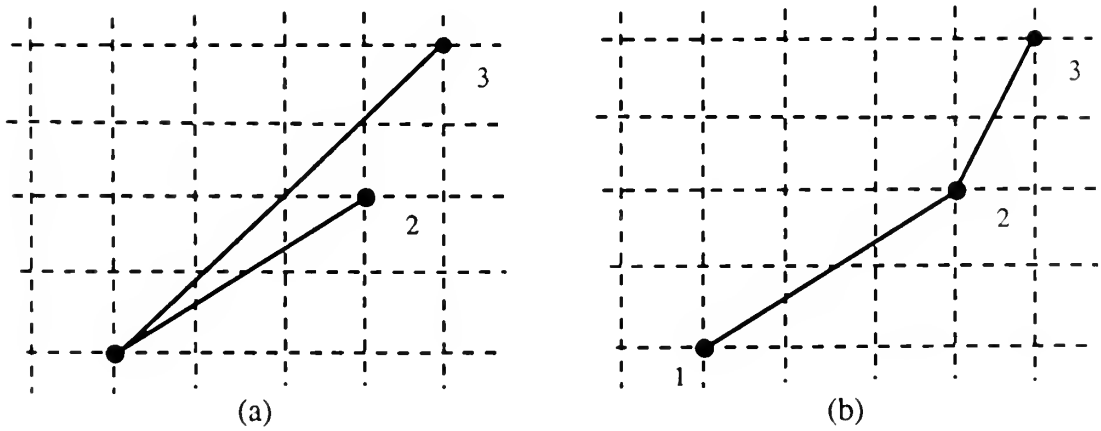


Figure 16. Original Flows and New Flows

Table 1. Empirical Results

1	2	3	4	5	6	7	8
			UB				
S.F.#	UB	LB	Improved	Time	LP	#iter	%gap
8.8.1	63	56.50	63	0.327	63.00	185	0.00
8.8.2	52	47.83	51	0.495	49.00	119	4.08
8.8.3	65	53.75	65	0.424	64.99	255	0.02
8.8.4	85	68.83	85	0.391	82.00	225	3.66
8.8.5	56	45.58	56	0.199	55.00	173	1.82
8.12.1	34	29.80	34	0.39	34.00	284	0.00
8.12.2	57	44.70	55	0.695	54.00	444	1.85
8.12.3	87	59.68	87	0.501	82.00	476	6.10
8.12.4	46	40.17	46	0.54	46.00	365	0.00
8.12.5	66	46.82	66	0.706	59.00	355	11.86
8.16.1	56	39.92	52	0.589	51.00	582	1.96
8.16.2	88	55.78	82	0.991	79.00	1037	3.80
8.16.3*	80	45.50	80	0.25	77.00	495	3.90
8.16.4	64	47.45	64	0.427	58.00	374	10.34
8.16.5	80	51.97	80	0.951	78.00	396	2.56
10.10.1	74	60.20	74	0.345	72.00	325	2.78
10.10.2	61	48.33	61	0.738	58.99	518	3.41
10.10.3	66	46.83	66	0.498	62.00	493	6.45
10.10.4	73	53.33	73	0.746	73.00	463	0.00
10.10.5	84	68.08	84	0.596	84.00	190	0.00
10.15.1	103	62.80	79	1.65	79.00	974	0.00
10.15.2	93	59.33	93	1.147	92.50	3272	0.54
10.15.3	102	62.30	101	1.372	93.00	1979	8.60
10.15.4	85	54.93	74	1.04	69.99	1720	5.73
10.15.5	105	57.93	105	1.735	93.99	3011	11.71
10.20.1	97	56.87	92	1.549	88.99	6015	3.38
10.20.2	96	62.39	86	1.805	86.00	1364	0.00
10.20.3	110	62.38	106	1.971	97.00	2884	9.28
10.20.4	108	54.48	82	2.077	77.99	5736	5.14
10.20.5*	90	55.40	90	1.374	88.00	1282	2.27
12.12.1	88	65.63	77	0.947	75.00	385+	2.67
12.12.2	56	44.10	56	0.331	52.50	207+	6.67
12.12.3	69	53.42	66	0.946	65.00	285+	1.54
12.12.4	85	56.68	83	0.903	78.00	1061+	6.41
12.12.5	110	63.70	103	1.295	103.00	492+	0.00
12.18.1	91	55.64	84	2.169	76.00	751+	10.53
12.18.2*	94	66.74	94	0.55	92.00	680+	2.17
12.18.3*	87	51.91	87	0.45	84.00	855+	3.57
12.18.4	101	58.61	88	1.746	86.50	384+	1.73
12.18.5	89	53.87	87	2.7	85.00	606+	2.35
12.24.1	130	74.65	123	2.314	113.00	2082	8.85
12.24.2	109	61.66	109	2.945	97.00	2679	12.37
12.24.3	103	66.03	102	1.633	96.00	947	6.25
12.24.4	99	53.29	92	1.505	92.00	2787	0.00
12.24.5	111	70.37	99	2.031	99.00	3581	0.00
			Average	1.09		Average	3.92

+on SUN workstation using CPLEX

References

Chhajed D. and V. Chandru (1988), "Rectiliner Hull - Efficient Sets - Convex Hull: Relationship and Algorithms," RM. 88-23, School of Industrial Engineering, Purdue University.

Chhajed, D.(1989), "Manufacturing Systems Layout, Station Location, and Flow Network Design," Ph. D. Thesis, Purdue University.

Dror, M., B. Gavish, and J. Choquette (1988), "Directed Steiner Tree Problem on a Graph: Models, Relaxation, and Algorithms," Publication 589, Centre de recherche sur les transports, universite de Montreal, Canada.

Egbelu, P. J. and J. M. A. Tanchoco (1986), "Potential for Bi-Directional Guide-Path for Automated Guided Vehicle Based Systems," International Journal of Production Research 24, 1075-1097.

Erlenkotter, D. (1978), "A Dual-based Procedure for Uncapacitated Facility Location," Operations Research 26, 803-834.

Fisher, M. L. (1981), " The Lagrangian Relaxation Method for Solving Integer Programs," Management Science 27, 1-18.

Foulds, L. R.(1983), "Techniques for Facilities Layout: Deciding which Pairs of Activities Should be Adjacent," Management Science 29, 1414-1426.

Francis, R. L. and J. A. White (1974), "Facility Layout and Location: an Analytical Approach," Prictice-Hall.

Gary, M. R. and D. S. Johnson (1977), "The Rectilinear Steiner Tree Problem is NP-complete," SIAM Journal of Applied Mathematics 32, 826-834.

Gaskins, R. J. and J. M. A. Tanchoco (1987), "Flow Path Design for Automated Guided Vehicle Systems," International Journal of Production Research 25, 667-676

Gaskins, R. J., J. M. A. Tanchoco, and F. Taghaboni (1989), "Virtual Flow Paths for Free Ranging Automated Guided Vehicle Systems, " International Journal of Production Research 27, 91-100.

Geoffrion, A. M. (1974), "Lagrangean Relaxation for Integer Programming," Mathematical Programming Study 2, 82-114

Giffin, J. W., Ph. D. Dissertation (1984), University of Canterbury, Christchurch, New Zealand.

Gomory, R. E. and T. C. Hu (1961), "Multi-Terminal Network Flows", J. SIAM 9, 551-570.

Hicks, B. P. (1987), "But What About the Little Guy ?" IE News 21, 2-3.

Hillier, F. S. and M. M. Connors (1966), "Quadratic Assignment Problem and the Location of Indivisible Facilities," Management Science 13, 42-57.

Ho Jan-ming, G. Vijayan, and C. K. Wong (1990), "New Algorithms for Rectilinear Steiner Tree," to appear in IEEE Trans. CAD/ICAS.

Itai, A., C. H. Papadimitriou, J. L. Szwarcfiter (1982), "Hamilton Paths in Grid Graphs," SIAM Journal of Computing 11, 676-686.

Johnson, D. S., J. K. Lenstra, and A. H. G. Rinnoy Kan (1978), "The Complexity of the Network Design Problem," Networks 8, 279-285.

Koopmans, T. C. and M. Beckmann (1957), "Assignment Problems and the Location of Economic Activities," Econometrica 25, 53-76.

Kusiak, A. and S. S. Heragu (1987), "The Facility Layout Problem," European Journal of Operations Research 29, 229-251

Magnanti, T. L. and R. T. Wong (1984), "Network Design and Transportation Planning: Models and Algorithms," Transportation Science 18, 1-55

Maxwell, W. L. and J. A. Muckstadt (1982), "Design of Automated Guided Vehicle Systems," IIE Transactions 14, 114-124.

Maxwell, W. L. and R. C. Wilson (1981), "Dynamic Network Flow Modelling of Fixed Path Material Handling Systems," AIIE Transactions 13, 12-21

Montreuil, B. (1987), "Integrating Design of Cell Layout, Input/Output Station Configuration, and Flow Network of Manufacturing Systems", in C. R. Liu, A. Requicha, and S. Chandrasekar (eds.), Intelligent and Integrated Manufacturing Analysis and Synthesis, ASME Ped-Vol. 25, 315-326.

Montreuil, B and H. D. Ratliff (1988(a)), "Optimizing the Location of Input/Output Stations Within Facilities Layout", to appear in Engineering Costs and Production Economics.

Montreuil, B and H. D. Ratliff (1988(b)), "Utilizing Cut Trees as Design Skeletons for Facilities Layout", to appear in IIE Transactions.

Montreuil, B and U. Venkatadri (1988), "From Gross to Net Layouts: An Efficient Design Model", Document No. 88-56, Operations and Decision Systems Department, Laval University, Quebec, Canada.

O'Brien, C. and S. E. Z. Abdul Barr (1980), "An Interactive Approach to Computer Aided Facility Layout, " International Journal of Production Research 18, 201-211.

Sahni, S. and T. Gonzalez (1976),"P-Complete Approximation Problems," J. ACM. 23, 555-565.

Tompkins, J. A. and J. A. White (1984), Facilities Planning, John Wiley & Sons.

Warnecke, H. J. and W. Dangelmaier (1983), "Progress in Computer Aided Plant Layout," Technical Report, Institute for Manufacturing Engineering and Automation - Fraunhofer, Stuttgart, Fed. Rep. of Germany.

Appendix 1. Formal Statements of Algorithms

ALGORITHM PREPROCESS

For each flow $f=(s_u, s_v) \in F$ do

if $hl(s_u) = hl(s_v)$ or $vl(s_u) = vl(s_v)$ then

fix arcs in S(f, G, 1);

$F = F \setminus f$;

else if R(f,G) has a unique path in G between s_u and s_v then

fix the path and set

$F = F \setminus f$.

{S(f,G,1) is the set of arcs on the lcr-path in R(f,G) with weight 1 on each arc.}

We note that after preprocessing, if $(s_u,s_v) \in F$, then $p(s_u) \neq p(s_v)$ and $q(s_u) \neq q(s_v)$.

ALGORITHM BUILD

Given: Flow set F, station set S, network G.

Objective: To find a good SRFN.

Step 0: For each $a \in E(G)$ do

$p_a = 0$.

{Initialize the potential of all arcs to zero.}

Step 1: For each flow $f \in F$ do

For each $a \in E(R(f,G))$ and (arc a yellow) do

$p_a = p_a + 1$.

{For all the arcs which are in the rectangle formed by the flow and which are also yellow increment the potential by 1.}

Step 2: For all ($a \in E(G)$) and (a yellow) do

$c_a = C_a/p_a$;

For all red arcs do

$c_a = 0$.

{For the yellow arcs in G set the weight equal to the length of the arc divided by its potential. For the red arcs, set the weights to zero.}

Step 3: Set $N = \phi$;

For all $f \in F$ do

$N = N \cup$ all $(S(f,G,\underline{c}))$.

{For each flow find all the lcr-paths using weights $\underline{c} = \{ c_a: a \in G\}$ and add them to the network N.}

Step 4: Return N. Stop.

ALGORITHM PATHIMPROVE

Given: N (a feasible solution) and the flow set F.

Objective: To improve the given solution by examining flows

Step 1: Set F' = F.

Step 2: While $F' \neq \phi$ do

Select $f = (s_u, s_v) \in F'$;

Paint CA(f, N) red;

Set cost of CA(f, N) as zero;

If there exists an r-path from s_u to s_v in R(f, N) consisting of only red arcs, then F = F\f;

F' = F'\f.

Step 3: Return N, F.

{CA(f, N) is defined to be those yellow arcs such that the removal of any one of them disconnects s_u from s_v in the graph R(f,N).}

ALGORITHM ARCIMPROVE

Given: A flow set F, Feasible SRFN N .

Objective : To improve the current solution by examining arcs.

Step 1: For each $f \in F$ do

For all ($a \in$ Pathset(f, N)) and (a yellow) do

$p_a \leftarrow p_a + 1$.

{Each flow contributes to the potential of an arc if there exists a shortest rectilinear path in N using the arc.

Pathset(f, N) consists of all the arcs in R(f,N) which are in some r-path between s_u and s_v of flow f.}

Step 2: $N \leftarrow N \backslash \{a: a \in E(N)$, a yellow and $p_a = 0\}$;

{Arcs which are not not in any r-path of any flow are deleted.}

Step 3: Let a' = $\{a: \min_{a \in E(N)} (p_a)$ and a yellow$\}$;

If a' = ϕ go to Step 7

else go to Step 4.

{Choose a yellow arc with minimum potential. If no such arc exists, Stop.}

Step 4: If for each $f \in F$, S(f, N\a, 1) exists, then go to Step 5

else go to Step 6.

{Check whether there exists an r-path for each flow after deleting a.}

Step 5: Set $N \leftarrow N \backslash a$. Go to Step 1.

{Delete arc a and go to Step 1.}

Step 6: Fix arc a. Go to Step 3.

Step 7: N is the improved solution. Stop.

Appendix 2. Finding Shortest Path in a Grid Graph

Problem: Given a grid graph G with weights on each arc and two diagonally opposite vertices (A,B) defining G, find a least-cost r-path between A and B in G.

We assume that G is rectangular as shown in Figure A1 (otherwise add missing arcs and vertices and place a very high cost on these arcs) and that vertex A is the South-East vertex and B is the North-West vertex. Let there be M vertical lines and N horizontal lines in G.

We create a directed grid graph D by directing the arcs in G as shown in Figure A2. An r-path in G is the same as a directed path in D from A to B. Number the nodes from 1 to MN, beginning with vertex A as 1 and moving from left to right and bottom to top (see Figure A2). Define sets $I_1, I_2, \ldots, I_{M+N-1}$, recursively as follows:

$I_1 = \{1\}$

$I_i = \{n: n \in V(D)$ and $\exists$ a node $n' \in I_{i-1}$ such that there is a directed arc from n' to n in $E(D)\}$, for $i = 2,3,\ldots,M+N-1$.

Let w_{ij} denote the weight on the arc from node i to node j. We now give a recursive (Dynamic Programming) method to find the shortest directed path from A to B in D. Let $f_i(k)$ denote the shortest directed path from A to $k \in I_i$.

ALGORITHM LEASTCOST

$$f_1(1) = 0 \qquad (1)$$

$$f_i(k) = \min \{f_{i-1}(k-1) + w_{k-1,k},\ f_{i-1}(k - M) + w_{k-M,k}\}, \qquad (2)$$

where $f_{i-1}(k-1)$ and $w_{k-1,k}$ are not defined if $k-1 \notin I_{i-1}$. Similarly for $f_{i-1}(k - M)$ and $w_{k-M,k}$. We find $f_i(k)$ for all $k \in I_i$ after finding $f_{i-1}(k')\ \forall\ k' \in I_{i-1}$. $f_{N+M-1}(NM)$ is the desired solution.

The correctness of the above algorithm is evident from the fact that a directed path from A to B in D passes through exactly M + N -1 grid points (including A and B) and uses exactly one grid point in each set I_i. There are at most two arcs, $a_{k-1,k}$ and $a_{k-M,k}$ incident to any vertex k (provided these arcs exist). Once optimal solutions to reach vertices

$a_{k-1,k}$ and $a_{k-M,k}$ are known, the optimal path to vertex k can be computed by using (2).

To show the complexity of this procedure, we note that every node is examined only once and at every node at most two additions and a comparison are done. Hence the complexity of the algorithm is O(MN) which is linear in the number of nodes in G.

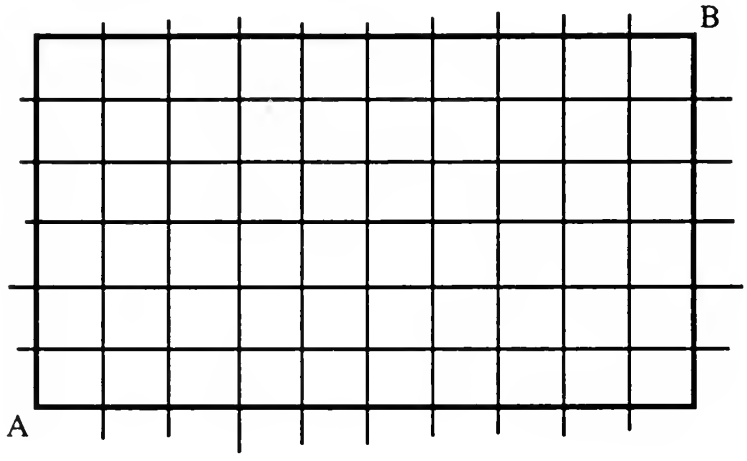
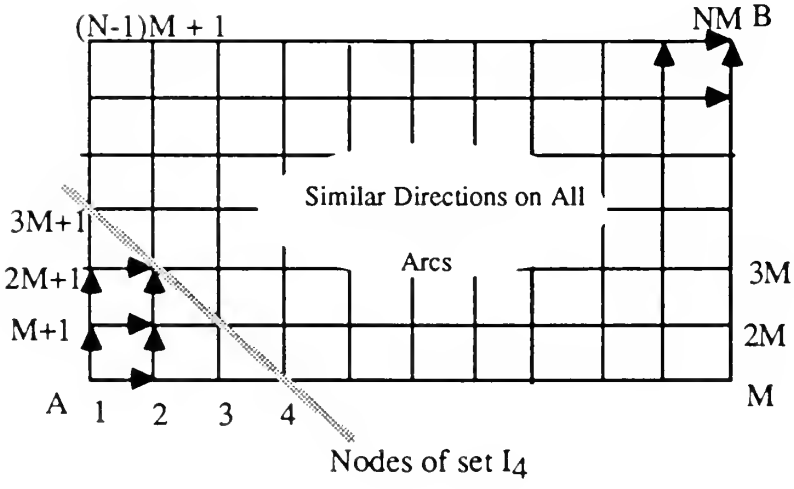


Figure A1
The Grid Graph for a Flow From A to B



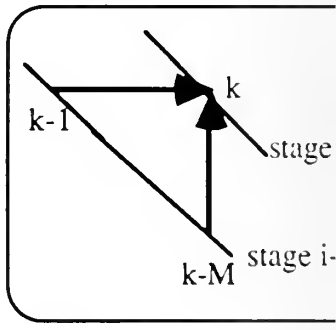


Figure A2
Grid Graph With Directions and Details of One DP Calculation