A Space Partitioning Method Embedded in a Simulated Annealing Algorithm for Facility Layout Problems with Shape Constraints

Jae-Gon Kim and **Yeong-Dae Kim**

Department of Industrial Engineering
KAIST, Yusong-gu, Daejon 305-701, Korea

Abstract

We deal with facility layout problems with shape constraints. A simulated annealing algorithm is developed for the problems. In the algorithm, a solution is encoded as a matrix that has information about relative locations of the facilities in the floor. A block layout is constructed by partitioning the floor into a set of rectangular blocks according to the information while satisfying areas of facilities. In this paper, three methods are suggested for the partitioning procedure and they are employed in the simulated annealing algorithm. Results of computational experiments show that the proposed algorithm performs better than existing algorithms, especially for problems with tight shape constraints.

1. Introduction

Generally, the problem of designing a physical layout of facilities with the objective of minimizing the total material handling costs is called the facility layout problem.

A major drawback of most of existing algorithms is that facilities with irregular shapes may appear in the final layout. This is because shapes of the facilities are not considered and layouts are developed on a floor which is divided into squares or rectangles with a unit area by a grid. A comprehensive survey of various algorithms for facility layout problems can be found in Kusiak and Heragu (1987). Recent efforts are placed on development of layout on a so-called *continual plane*, a floor which is not divided into unit areas by a grid (Tam 1992, Tate 1995).

This paper focuses on the facility layout problems with shape constraints. In this study, the block layout (floor plan) is constructed on a continual plane, and hence the facilities may have real values for the lengths and widths. Facilities are modeled as rectangles in the block layout and the shape constraint for each facility is represented by the *aspect ratio*, that is, the aspect ratio of each block must be within a given range. Here, the aspect ratio is defined as the ratio of the maximum of the length and the width of the facility to the minimum of them.

2. Representation of the Solutions

In the suggested algorithm, a solution is represented by an $r \times c$ matrix **M**, which has information about relative locations of the facilities in the floor. The elements of the matrix specifies the indices of the facilities. The number of elements, rc, is greater than or equal to the number of facilities, N. When rc is strictly greater than N, $rc - N$ dummy facilities are introduced. Dummy facilities have neither area nor material flow from/to other facilities and are indexed zero. Let m_{ij} be the value of the element at row i and column j in **M**. Then, by the encoding scheme suggested here, facility m_{ij} is to be placed to the right of facility $m_{i-1,j}$, to the left of facility $m_{i+1,j}$, below facility $m_{i,j-1}$, and above facility $m_{i,j+1}$ in the floor. In this manner, relative locations of all facilities in the floor are determined by **M**. In this paper, **M** is called the *location matrix*.

To represent a solution with **M**, r and c must be determined. Let L and W be the length and the width of the floor, respectively. In the suggested scheme, r and c are determined with $r = \lceil \sqrt{NL/W} \rceil + k_r$, and $c = \lceil \sqrt{NW/L} \rceil + k_c$, where $\lceil a \rceil$ denotes the smallest integer which is greater than or equal to a, and k_r and k_c are parameters with non-negative integer values. If k_r and k_c are set to zero, r and c are determined in such a way that their ratio is proportional to that of L and W. As k_r and k_c become greater, more dummy facilities are introduced. These two parameters, k_r and k_c are determined by another parameter k with a positive integer value as follows.

If $\left| \lceil \sqrt{NL/W} \rceil - \sqrt{NL/W} \right|$ is less than $\left| \lceil \sqrt{NW/L} \rceil - \sqrt{NW/L} \right|$, let $k_r = (k+1)/2$ and $k_c = (k-1)/2$,

otherwise, let $k_r = (k-1)/2$ and $k_c = (k+1)/2$.

As k increases, k_r and k_c become larger and as a result, the number of alternatives for the block layouts increases. In general, k need not be too large unless there are extremely small or large facilities or extremely long or wide facilities.

3. Space Partitioning Method

In this section, we suggest a method called the *space partitioning method* (SPM), which decodes a location matrix into a block layout. In the suggested method, the location matrix is decomposed recursively and then the floor is partitioned according to the results of the decomposition of the location matrix. The method partitions the floor into rectangular blocks using guillotine cuts.

In the suggested method, decomposing the location matrix into two submatrices corresponds to partitioning the floor into two blocks. If the location matrix is cut horizontally (vertically) and decomposed into an upper (left) submatrix and a lower (right) submatrix, the floor is partitioned into an upper (left) block and a lower (right) block. The floor is partitioned by a guillotine cut in such a way that the area of each block is equal to the sum of the areas of the facilities of which the indexes are included in the submatrix corresponding to the block.

If the above procedure is recursively applied to the submatrices and partitioned blocks, the location matrix is decomposed down to rc 1×1 matrices (or elements), and the floor can be partitioned into rc blocks. (Once the location matrix is decomposed down to element levels, a complete layout can be constructed using SPM.) Note that blocks corresponding to $rc-N$ elements with 0 values have no area and hence they do not appear in the layout.

Heuristic methods are used for decomposing a location matrix. A layout generated by SPM satisfies area constraints for all facilities but does not always satisfy shape constraints. Because of this, the heuristic methods have a surrogate objective of minimizing the number of facilities of which the shape constraints are violated in the final layout or minimizing the probability that the shape constraints are not satisfied. In this study, three methods are developed.

Method A

In this method, two alternatives are considered for decomposition. The first alternative is to decompose the location matrix into row vectors and then decompose them into elements. The second alternative, on the other hand, is to decompose the location matrix into column vectors and then decompose them into elements. Note that once the location matrix is decomposed into row or column vectors, a unique final layout is obtained by SPM and one can easily find the number of facilities of which the shape constraints are violated. The alternative selected for the final layout is the one with the smaller number of such facilities.

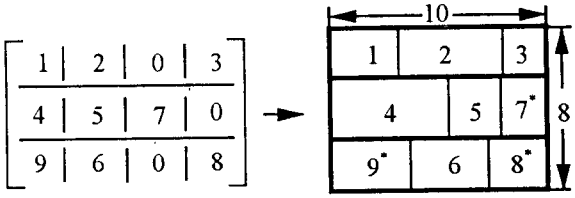
This method is illustrated with an example problem with $N = 9$, $L = 8$ and $W = 10$. Areas and ranges of aspect ratios of the facilities are given in Table 1 and Table 2. Figure 1 shows the two alternatives for decomposition and their corresponding layouts. The facilities marked with asterisks in the figure are those of which the shape constraints are violated. In this example, alternative (a) is selected since it has a smaller number of unsatisfied shape constraints.

Table 1. Areas of facilities in the example problem

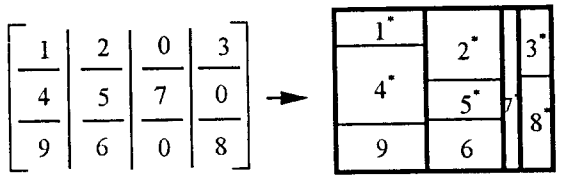
Facility	1	2	3	4	5	6	7	8	9
Area	8	12	5	16	7	9	6	7	10

Table 2. Aspect ratios of facilities in the example problem

	Facility								
	1	2	3	4	5	6	7	8	9
Lower bound	1.0	1.7	1.0	1.6	1.0	1.0	1.0	1.3	1.6
Upper bound	1.4	2.0	1.5	2.1	1.3	1.5	1.3	1.5	2.2



Alternative a



Alternative b

Figure 1. Method A

Method B

In this method, the location matrix is decomposed into two submatrices one of which is a column or a row vector. There are four alternatives for such decomposition: to select the first row, the last row, the first column and the last column of the matrix as one of two submatrices. For each alternative, an upper bound on the number of

facilities of which the shape constraints are violated can be obtained by applying Method A to the submatrices. The method selects the one which gives the minimum upper bound among these four alternatives. This procedure is applied to the submatrices until the location matrix is decomposed down to row or column vectors. Once the location matrix is decomposed into row or column vectors, a complete block layout can be easily obtained.

Figure 2 shows how Method B is applied to the example given above. In the resulting layout, there are two facilities of which the shape constraints are not satisfied.

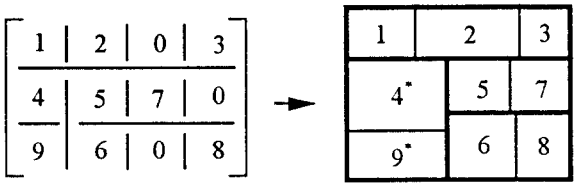


Figure 2. Method B

Method C

In this method, every possible alternative for guillotine cuts is considered for decomposition. For example, if there are r rows and c columns in the matrix, there are $r+c-2$ ways to decompose the matrix into two submatrices. The method selects the one which gives the minimum upper bound on the number of facilities of which the shape constraints are violated. The upper bound is obtained with the same method as in Method B. This procedure is recursively applied to submatrices resulted from such decomposition until the location matrix is fully decomposed down to row or column vectors.

The previous example problem is solved by this method as well, and the result is given in Figure 3. In this example problem, this method gives a feasible block layout in which shape constraints of all facilities are satisfied.

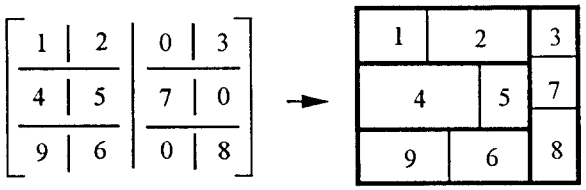


Figure 3. Method C

4. The Simulated Annealing Algorithm

In this study, an SA algorithm is used to find a location matrix which gives the best layout. For an implementation of an SA algorithm on the problem considered here, we determine various parameters or methods as follows.

Solution space

As explained earlier, a solution is represented by an $r \times c$ matrix. Note that only N of rc elements of the matrix are assigned to N facilities and the rests are allocated to dummy facilities. A parameter (k) is used to control the size of the matrix or equivalently, the size of solution space (see section 3).

Objective function

In the suggested SA algorithm, the shape constraints are converted into a penalty so that feasible solutions are obtained more easily in the search procedure. That is, the objective function value of a solution in the SA algorithm is computed with $T_D \cdot (1+\alpha N_s)$, where T_D is the solution value (total travel distance) obtained from the given layout, N_s is the number of facilities of which the shape constraints are not satisfied and α is a penalty weight for N_s. After a series of preliminary experiments, this weight is set 0.3 in the suggested SA algorithm.

Neighborhood generation

A neighborhood solution of the current solution is generated by exchanging two elements in the location matrix corresponding to the current solution.

Cooling schedule

In the suggested algorithm, the initial temperature T_0 is chosen in such a way that the fraction of accepted uphill moves in a trial run of the annealing process is approximately F_0. For the algorithm, N moves are made and the average increase in the objective function value $\overline{\Delta}$ is calculated with uphill moves only and then T_0 is obtained from the equation, $\exp(-\overline{\Delta}/T_0) = F_0$. The epoch length specifies the number of moves made with the same temperature. In the suggested algorithm, the epoch length is set to be $l\binom{rc}{2}$, where l is a parameter to be determined. Also, the temperature is decreased in such a way that the temperature at the k-th epoch is given by $T_k = \gamma T_{k-1}$, where γ is a parameter, called the cooling ratio, with a value less than 1.

Stopping condition

As the criterion for termination, the suggested algorithm adopts the method given by Johnson *et al.* (1989). The method maintains a counter that is incremented by one when an epoch is completed without any improvement in the solution value and that is reset to 0 when a new incumbent solution is found. The SA algorithm is terminated when the counter reaches a given limit. After a series of tests on a number of problems, 3 is selected for the limit in the suggested algorithm.

5. Computational Experiments

The solution encoding scheme with three methods suggested for decomposition of the location matrix was implemented in the SA algorithms. These SA algorithms are denoted by SA-A, SA-B and SA-C according to the methods. They are compared with a algorithm of Tate and Smith (1995) denoted by T&S that is known to give better solutions than others. The suggested SA algorithms were coded in C language and all the algorithms included in the tests were run on a personal computer with a Pentium processor.

In general, layout problems with shape constraints can be characterized by the number of facilities, tightness of shape constraints, and variance of areas of the facilities, among others. We generated 10 problems for each of all combinations of four levels for the number of facilities (10, 15, 20, and 30), two levels for the tightness (loose and tight) and two levels for the variance (small and large).

After a series of experiments, it was found that the SA algorithms with $k = 1$, $F_0 = 0.65$, $\gamma = 0.85$ and $l = 2$ gave better solutions than others without requiring much longer computation time, so they were selected for the algorithms. Parameters for T&S were set as in Tate and Smith (1995) except for the stopping condition. As the stopping criterion for T&S, we used the number of iterations for which the best solution has not been improved. When the number reaches a given limit, C_N, the algorithm is terminated. The limit C_N was set to be equal to $1600N^2$ in T&S.

Tables 3 and 4 show results of the tests on the three SA algorithms and T&S. In Table 3, the relative deviation percentage is defined as $100(F_a - F_B)/F_B$ for algorithm a, where F_a and F_B are the solutions from algorithm a and from the algorithm which gave the best solution for the problem, respectively. Problems for which an algorithm did not find a feasible solution are excluded from the calculation of this percentage for the algorithm.

Table 3. Performance of the algorithms

	T&S	SA-A	SA-B	SA-C
NFS	95	105	148	158
NBS	1	5	59	105
RDP	9.12	6.36	1.61	0.84

NFS: number of problems for which each algorithm found a feasible solution

NBS: number of problems for which each algorithm found the best solution

RDP: relative deviation percentage

Table 4. Average CPU time (in seconds) for the algorithms

N	T&S	SA-A	SA-B	SA-C
10	33.42	12.49	43.95	56.13
15	127.62	41.85	158.71	238.59
20	348.87	83.23	371.01	651.10
30	1548.86	398.87	1710.80	2872.67

The suggested SA algorithms outperformed the existing algorithm. SA-A slightly outperformed T&S in the solution quality and the computation time, and SA-B gave much better solutions than T&S in almost the same computation time. SA-C gave the best solutions although it required the longest computation time. Since SA-C employs a solution decoding scheme which checks a large number of alternatives for the block layout, it can find feasible layouts more easily.

6. Concluding Remarks

In this paper, a simulated annealing algorithm was applied to the facility layout problem with shape constraints. A new layout representation scheme was developed for solution encoding. We suggested the space partitioning method (SPM) to convert the encoded solution into a layout. Results of comparisons with existing algorithms showed that the suggested SA algorithms outperformed the existing ones, especially for problems with tight constraints.

This research can be extended in several ways. For example, other methods can be used to determine the size of the location matrix and/or to decompose the location matrix. Also, other search methods such as tabu search and genetic algorithms can be applied to the problems even with the same encoding/decoding scheme.

References

Johnson, D., Aragon, C., McGoech, L. and Schevon, C. (1989) Optimization by simulated annealing: an experimental evaluation; part I, graph partitioning. *Operations research*, **37**(6), 865-892

Kusiak, A. and Heragu, S.S. (1987) The facility layout problem. *European Journal of Operational Research*, **29**, 229-251.

Tam, K.Y. (1992) A simulated annealing algorithm for allocating space to manufacturing cells. *International Journal Production Research*, **30**(1), 63-87.

Tate, D.M. and Smith, E.A. (1995) Unequal-area facility by genetic search. *IIE Transactions*, **27**(4), 465-472.