FACOPT: A User Friendly FACility Layout OPTimization System

Jaydeep Balakrishnan[1]

Chun-Hung Cheng[2]

Kam-Fai Wong[2]

Published in *Computers and Operations Research*, 30, 2003, pp 1625-1641.

1. Faculty of Management, University of Calgary, Calgary, Alberta T2N 1N4, Canada

2. Department of Systems Engineering and Engineering Management, The Chinese University of Hong Kong, Shatin, Hong Kong.

Corresponding Author:
Jaydeep Balakrishnan
Faculty of Management
University of Calgary
Calgary, Alberta T2N 1N4 CANADA
Ph: (403) 220 7844 Fax: (403) 284 7902 Internet: balakris@ucalgary.ca

FACOPT: A User Friendly FACility Layout OPTimization System

Abstract

The facility layout problem is a well researched one. However few effective and user friendly approaches have been proposed. Since it is an NP hard problem, various optimization approaches for small problems and heuristic approaches for the larger problems have been proposed. For the most part the more effective algorithms are not user friendly. On the other hand user friendly methods have not been effective in handling the intricacies such as unequal department sizes. In this research we present FACOPT, a heuristic approach that is effective and user friendly. The software uses two methods, Simulated Annealing and Genetic Algorithm to solve the facility layout problem. Computational tests are also done to identify good parameter values and to compare the performance of the two algorithms.

Scope and Purpose

Various methods have been proposed for facility layout where departments are laid out within a facility. However many of them are not flexible enough to handle intricacies such as unequal department sizes. Others do not provide user-friendly interfaces. Thus there is a need for user-friendly software incorporating effective and flexible procedures. In this research we present FACOPT, a heuristic approach that is effective and user friendly. The software uses two different approaches to solve the facility layout problem effectively. FACOPT has a Visual BASIC interface and runs under a Windows environment for ease of use.

Keywords: Facility Layout, Simulated Annealing, Genetic Algorithms, Software, User -friendly

FACOPT: A User Friendly FACility Layout OPTimization System

Introduction

The static plant layout problem (SPLP) is a well-researched problem. Koopmans and Beckman [1957] first modeled it as a quadratic assignment problem (QAP) because the objective function is a second-degree function of the variables while the constraints are linear.

The SPLP is a difficult problem to solve optimally due to its structure. Hence, heuristic algorithms have been proposed to solve the problem. Many of these can handle only equal sized departments but others can handle the more practical case where the department sizes are unequal. In this paper we assume that department sizes are unequal.

Two methods, simulated annealing and genetic algorithms are used to solve the SPLP. The software described here, FACOPT, can use either algorithm to obtain a good solution. In addition, FACOPT has the following features that make it of much practical use to the decision-maker:

- FACOPT can handle facilities with large numbers of departments. The test problems had as many as 30 departments
- In FACOPT, the user may employ more than one algorithm and specify parameter values other than default values for a chosen algorithm.
- FACOPT retains ten layout solutions in each session. The user can switch from one to another and compare different layouts.
- FACOPT accepts both manual entry and automatic generation of space-filling curves. The concept of space-filling curves will be discussed later in this paper.

- FACOPT allows users to fine-tune a layout by forcing the exchange of the department locations.
- FACOPT permits fixed departments and is flexible enough to consider other layout design considerations.
- The user interface in Visual Basic for Windows with its ability to use color allows for ease of use.

The Static Plant Layout Problem

A rectangular layout with six locations is shown in Figure 1. The locations are all equal in size. There are six departments (1 though 6) that have to be assigned to the six locations. One such assignment is shown in the figure.

1	**2**	**3**
4	**5**	**6**

Figure 1: An example layout

This is only one out of the 6! or 720 combinations that exist for this layout. Each combination represents a different static layout (though in this particular example due to symmetry there are only 120 unique layouts). There is material handling flow between the different departments and a cost associated with the unit material handling flow per unit distance. Thus different layouts can have different total material handling costs

depending on the relative location of the departments. The mathematical programming formulation (QAP) for this problem is as follows:

$$\text{Min} \quad Z = \sum_{i=1}^{n} \sum_{j=1}^{n} \sum_{k=1}^{n} \sum_{l=1}^{n} f_{ik} d_{jl} X_{ij} X_{kl} \tag{1}$$

$$s.t.$$

$$\sum_{i=1}^{n} X_{ij} = 1, \; j = 1,\ldots,n \tag{2}$$

$$\sum_{j=1}^{n} X_{ij} = 1, \; i = 1,\ldots,n \tag{3}$$

where

n : Number of departments in the layout
i,k : Departments in the layout
j,l : Locations in the layout
f_{ik} : Flow cost from department i to k
d_{jl} : Distance from location j to l
X_{ij} : 0,1 variable for locating department i at location j

Equation (1), the objective function minimizes the sum of the flow cost over every pair of departments. Equation (2) ensures that each location contains only one department. Equation (3) ensures that each department is placed only at one location.

The problem investigated in this paper is an extension of this problem. In the QAP each department is assumed to be of equal size (equal sized layout). In practice departments may be of varying sizes (unequal sized layout). The unequal sized layout problem may be modelled as a QAP problem by dividing the areas into a grid with equal sized squares. Each department would occupy different numbers of squares depending on its size. This is seen in Figure 2 where department 2 occupies squares B, C and E while department 1 occupies squares A and D. The shapes of the

departments can be made more accurate by using smaller grids. But this will further increase the size of the problem and the resultant computation time.

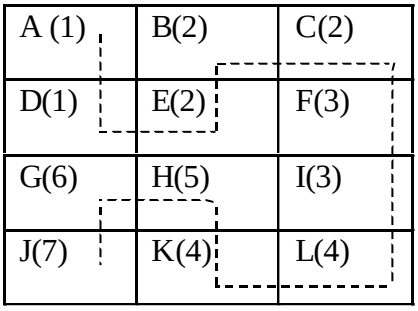
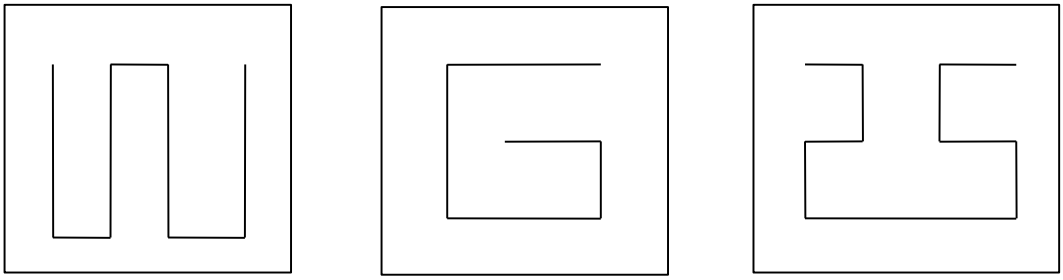


Figure 2: An example of a twelve square layout and a SFC

One problem in modeling the problem in this manner is to ensure that the all the squares representing one department remain contiguous when the algorithm is searching for solutions. This is done using space-filling curves (SFC). These were developed by Bartholdi and Platzman [1982] for the travelling salesman (TSP) and shortest path problems. It was extended to facility layout by Bozer et al.. These curves are formed by connecting a series of Hilbert curves. Kochhar et al. point out that these curves can be generated only in the absence of fixed departments, blocked areas, or irregularities. However, psuedo-Hilbert curves may be drawn by hand (Figure 3). Hand drawn curves can follow a sweeping pattern, spiraling pattern or a space-filling pattern producing different final solutions.



(a) (b) (c)

Figure 3: Psuedo-Hilbert curve

We now demonstrate how SFCs can be applied. Figure 2 shows a grid with 12 locations (squares), A through L. The SFC forms a continuous curve that traverses through all the squares in the grid in such a way that it goes through neighboring squares before moving to non-neighboring ones. If departments are located along neighboring locations on the SFC, we are guaranteed a feasible solution where the locations that contain the same department are contiguous. For example in Figure 2, department 1 is located at A and D which are neighboring squares on the SFC.

Literature Review

In addition to the QAP, the layout problem has been modeled as a quadratic set-covering problem (Bazaraa [1975]), linear integer programming (Lawler [1963]), mixed integer programming (Kaufman and Broeckx [1978]), and a graph theoretic problem (Leung [1992]) among others. Sahni and Gonzalez ([1976]) showed that the QAP is NP complete. Thus the largest problem solved optimally contained only fifteen departments (Kusiak and Heragu [1987]). Thus, problems with more than twenty facilities are unlikely to be solved optimally. The main types of algorithms

used to solve the problem optimally may include branch-and-bound, and cutting plane algorithms.

Since optimal methods are limited in the number of departments that can be included, substantial research has be done in developing sub-optimal algorithms that can solve larger problems. These are basically of two types: construction algorithms where a layout is constructed from scratch, and improvement algorithms where an existing solution is improved. HC66 (Hillier and Connors [1966]) is an example of a construction algorithm. CRAFT (Armour and Buffa [1963]) is an early example of a improvement algorithm while HOPE (Kochhar et al. [1998]) is a recent one. SPACECRAFT (Johnson [1982]) is one of the earlier examples of a three-dimensional improvement algorithm, which can handle multiple floors. MULTIPLE (Bozer et al. [1994]) and MULTI-HOPE (Kochhar and Heragu [1998]) are some recent examples of multiple floor layout algorithms. Researchers have also applied the more recent search techniques, which have proved to be effective, such as simulated annealing (Jajodia et al. [1992]), neural networks (Tsuchiya et al. [1996]), genetic algorithms (Kochhar et al.), and tabu-search (Skorin-Kapov [1994]). Detailed reviews of the facility layout literature can be found in Kusiak and Heragu [1987] and Meller and Gau [1996]. However, many of these procedures are difficult to use since the software implementation is not user friendly. In this paper we propose a user-friendly algorithm that uses some of the effective techniques that exist in the literature.

FACOPT's Algorithms

FACOPT considers two of the more recent search algorithms: simulated annealing and genetic algorithms. In the computational studies found in the literature,

these have proved to be among the more effective. In this section, we briefly describe these algorithms and discuss the determination of various parameter values.

Simulated Annealing (SA)

Kirkpatrick et al. [1982] introduced the concepts of SA in combinatorial optimization. An excellent discussion on the use of SA in Operations Research can be found in Eglese [1990]. SA is a type of local search algorithm. In the facility layout problem, a particular solution can be likened to the properties of a solid at a temperature, T. A lower energy state would be a lower cost solution. Thus 'cooling' is the process of obtaining better solutions. The process starts with a relatively high value of T (which in facility layout would be a control parameter) to avoid being prematurely stuck in a local optimum. The SA algorithm then performs a neighborhood search at each temperature and the temperature is gradually lowered. One of the features of SA is that during the cooling process occasionally a poorer solution than the existing one may be accepted. This may prevent the procedure from being trapped by local optima. In such a procedure the poorer the solution, the less likely it is to be chosen. Also as T decreases, the probability of accepting poorer solutions decreases.

Our SA is similar to SABLE (Meller and Bozer [1996]). The pseudo-code in Figure 4 is adopted from Tompkins et al. [1996]. Specifically, SA takes a layout and perturbs it randomly into a new layout in the neighborhood of the original one if the new layout has a lower cost. As mentioned, a poorer layout may be retained. The algorithm moves to a lower temperature state when the mean improvement it finds between a neighbourhood group (epoch) searched and the overall mean is less than a threshold value for a temperature state. The epoch length denotes the maximum

number of accepted moves made in a neighbourhood that forms a group for calculation of the mean improvement. The SA terminates in two ways: either when a pre-specified number of successive temperature reductions without improvement is reached, or when the total number of epochs searched reaches the pre-defined limit.

Step 1.

Create the initial layout sequence randomly as the current sequence *s* and compute the layout cost *f(s)* using a given SFC and the size requirements of departments. Set the temperature counter $i = 1$ and temperature is t_i .

Step 2.

Randomly choose two departments in the current sequence *s*, exchange their position in the sequence. Store the resulting sequence as the candidate sequence *s'*.

Step 3a.

Compute the decrease in the layout cost, i.e., $\Delta f = f(s) - f(s')$. If $\Delta f > 0$, go to step 3c.

Step 3b.

Accept this non-improving layout by going to step 3c with probability $exp\ (\Delta f / t_0(\alpha)^i)$, where t_0 is the initial temperature, α is the temperature reduction factor, and ; otherwise, go to step 2 to generate another one.

Step 3c.

Accept the candidate sequence and also take it as the current sequence. If the cost for the current sequence is lower than that of the "current best" sequence, update the "current best" sequence. If *e* candidate layouts have been accepted, go to step 4; otherwise, go to step 2, where *e* is the fixed epoch length. Let mf_e be the mean cost of the e layouts accepted in this epoch. Let mf_a be the mean cost of the layouts accepted previous to this epoch.

Step 4.

If $| mf_e - mf_a | / mf_a => \epsilon$, equilibrium has not been reached at temperature t_i; reset the counter for accepted candidate solutions and go to step 2 to generate *e* more accepted solutions. Otherwise, set $i = i + 1$ and update $t_i = t_1 \alpha^{i-1}$; if the maximum number of successive non-improving temperatures has been reached, STOP, otherwise go to step 5.

Step 5.

If the total number of epochs examined has reached its pre-defined limit, STOP; otherwise, go to step 2.

Figure 4: A SA pseudo-code

In SA, the user is required to specify several parameters. A set of parameters must be empirically found to substantially reduce the initial layout bias and generally generate a "good" solution. Based on Tompkins ET al. [1996], and Kim and Kim [1998], the number of choices was narrowed down. In addition, testing for parameter values was done using the Tam _30 problem set (Tam [1992]). In Tables 1-3, two of the six parameters are varied while holding the others constant. As seen in the three tables, this involved a total of 66 different setting. The solutions for different settings of the parameters are shown as percentages above the best solution. The parameter combination that gave the best solution was used as part of the fixed parameter setting in the subsequent tables. For example in Table 1, $\alpha = 0.90$ and $t_0 = 0.6$ gave the best solution of 48571.95. Thus this combination is part of the fixed parameters in Tables 2 and 3. Based on Tables 1-3, it was decided to use the following parameter values:

- temperature reduction factor: $\alpha = 0.90$
- initial temperature: $t_0 = 0.6$
- epoch length: $e = 30$
- equilibrium threshold value: $\varepsilon = 0.05$
- maximum number of epochs to be considered: $M = 37$
- maximum number of non-improving solutions: $N = 5$.

It is also seen that depending on the parameter setting, the solution may be as high as 8.5% above the best solution. Thus in practice, it might also be useful to run the solution with a few different parameter settings to examine whether better solutions can be obtained.

α / t_0	0.2	0.4	0.6	0.8
0.8	2.1%	1.7%	5.6%	5.6%
0.85	1.9%	1.1%	2.3%	7.6%
0.9	4.6%	2.9%	**0.0%**[1]	8.5%
0.95	4.6%	3.0%	1.5%	8.5%

[1] This setting provided the best value of 48571.95

Table 1: e=30, M=37, N=5, ε=0.05

e / M	e+3	e+5	e+7	e+9	e+11
20	8.1%	4.2%	4.3%	5.2%	1.9%
25	3.0%	7.3%	2.6%	5.8%	1.4%
30	5.1%	1.7%	**0.0%**[1]	5.9%	3.7%
35	8.1%	3.9%	7.7%	5.3%	4.1%
40	3.3%	7.8%	5.6%	6.0%	1.6%

[1] This setting provided the best value of 48571.95

Table 2: α=0.9, t_0=0.6, N=5, ε=0.05

ε / N	1	3	5	7	9
0.01	7.9%	1.5%	3.0%	6.7%	6.6%
0.03	7.2%	1.9%	6.3%	4.4%	4.6%
0.05	7.3%	2.2%	**0.0%**[1]	5.6%	5.6%
0.07	1.6%	6.7%	5.2%	7.3%	7.2%
0.09	7.9%	4.9%	5.9%	3.1%	4.3%

[1] This setting provided the best value of 48571.95

Table 3: e=30, M=37, N=5, ε=0.05, α=0.9, t_0=0.6

Genetic Algorithms (GA)

A GA based procedure (Holland [1975]) requires that an initial population of feasible solutions be generated. It codes a solution as a finite-length string composed of some finite alphabets. Each feasible solution may be a parent. The population is also known as the parent pool. The procedure uses a fitness function (that is constructed based on the layout objective function) to determine the fitness of each potential parent. It obtains better solutions through selection, reproduction, crossover, and mutation. Our GA code is given in Figure 5.

Step1.
Initialize the parent population pool.
Step 2.
Evaluate the fitness values of all individuals in the parent population pool.
Step 3.
Select two parents for the reproduction process (including crossover and possibly mutation) to produce their offspring.
Step 4.
Evaluate the offspring produced.
Step 5.
Replace the worst parent with the offspring produced.
Step 6.
If the pre-determined number of generations is not reached, go to Step 3; otherwise, STOP.

Figure 5. A GA pseudo-code

Typically, the parent population is composed of many individuals. Randomly generate department sequences form the parent pool. A department sequence, department sizes, and a pre-defined SFC are used to obtain a layout solution.

The fitness function determines the probability of an individual being chosen for reproduction. The fitness function used is the objective function of SPLP (Equation 1).

The roulette wheel method is used to select the parents for crossover and the crossover procedure is similar to the one found in HOPE (Kochhar et al.) Occasionally mutation is applied to the child layout after crossover. The mutation process introduces diversity. This helps prevent the algorithm from converging too quickly to a local optimum.

The new child can then be used to replace the parent with the highest cost in the parent pool. The GA is terminated when a specified number of generations is reached.

The basic operator for producing a child is crossover. It randomly selects one of the two parents as parent 1. Then, it randomly chooses a department from the department sequence of parent 1 and places it in the child department sequence in the same location as it existed in parent 1. This process is continued for *n/2* departments where *n* is the number of departments. The departments missing in the child's department sequence are filled in the same order as they appear in parent 2.

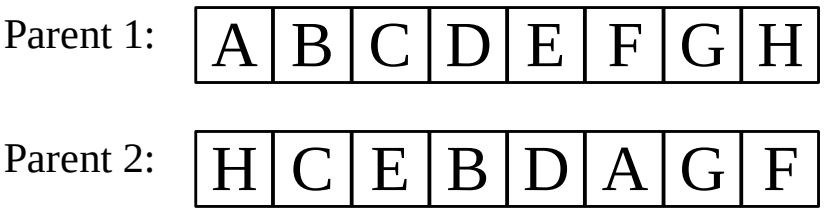


Figure 6. Two parents' department sequences for crossover

Consider the two parents in Figure 6. The letters in each parent represent a department sequence. To construct the child, four departments are randomly picked from parent 1 and placed in the new offspring. Assume departments A, C, D, and H are selected. The partial child's department sequence is shown in Figure 7. The missing departments in the department sequence are then filled in the same order as they appear in parent 2. The complete department sequence is shown in Figure 8.

Figure 7: Partial Child

Figure 8. A complete child

Mutation is a random process that is occasionally applied to introduce diversity. Our mutation procedure is simple. Two departments in a department sequence are swapped to produce a new genetic structure. The mutation rate is the pre-specified probability determining how frequently mutation takes place. The decision process includes two steps: (a) generate a random number, *r*, such that $0 \leq r \leq 1$ and (b) if the random number *r* is less than or equal to the pre-specified mutation rate, perform the mutation.

To set up the GA procedure, we will have to analyze the effect of mutation rates and population pool sizes for a given number of generations (iterations) on the quality of solutions. The Tam_30 data set is used. The summary of the computational results is given in Table 4 with the best result highlighted. The given number of generations was set as 4500 since we found that it was sufficient to achieve convergence. Based on this result, the following parameter values were selected:

- Population size, N = 35
- Mutation rate, P_m=0.09
- Number of generations = 4500.

P_m\N	15	20	25	30	35	40	45
0.002	7.1%	5.0%	7.3%	4.4%	4.1%	1.9%	2.9%
0.004	8.9%	5.0%	1.6%	2.5%	2.2%	2.4%	1.5%
0.006	5.9%	5.4%	4.0%	1.4%	3.8%	5.8%	3.1%
0.008	8.6%	4.1%	9.3%	7.4%	1.8%	2.4%	0.6%
0.01	9.3%	3.6%	8.7%	5.3%	4.4%	1.5%	4.3%
0.03	8.1%	2.9%	5.5%	5.1%	0.5%	3.4%	3.7%
0.05	6.6%	4.7%	3.8%	4.3%	1.1%	1.3%	3.3%
0.07	3.0%	1.6%	5.4%	0.7%	**0.0%**[1]	1.9%	2.0%
0.09	1.6%	3.5%	4.8%	4.8%	2.0%	3.8%	3.8%

[1] This setting provided the best value of 42638.63

Table 4: GA parameter selection

Again, it is seen that depending on the setting the solution can be as high as 9.3 above the best solution obtained. Thus in practice, it might be worthwhile solving the problem with a few different parameter settings to examine whether better solutions can be obtained.

The tests also show the advantage of having more than one method to solve the problem. Depending on the parameter setting, one method may do better or worse than another algorithm. For example when: e=30, M=37, N=5, ε=0.05, α=0.9 and , t_0=0.8, (Table 1) the SA method gives a value of 52700.22 which is almost 24% higher than the best GA solution of 42638.63 (Table 4). When e=30, M=37, N=5, ε=0.05, α=0.9 and , t_0=0.6, the SA method gives a value of 48571.95 which is only about 1% higher than the worst GA solution of 46604.29. Though the GA performed better than the SA in this problem set, given the wide variance between the solutions from each parameter setting, there is no guarantee that GA will perform better in every situation. Thus it is useful to have more than one method available. This aspect is also seen in our main experiment.

Since in the main experiment, we modified the original Kim and Kim [1998] problems, the Tam_30 data set (which was not modified) also serves as a benchmark for our procedure. The best solution obtained in Tam was 48072.92 which is about 13% higher than our best GA solution of 42638.63. However our best SA solution of 48571.95 was about 1% higher than the best Tam solution.

Computational Results

The 60 problems obtained from Kim and Kim [1998] were modified by restricting department sizes to an integral number of square units and making some problems feasible. The problem sizes examined are shown in Table 5.

Problem Numbers	**Departments**
1 – 5 16 – 20 31 – 35 46 - 50	10
6 – 10 21 – 25 36 – 40 51 - 55	20
11 – 15 26 – 30 41 – 45 56 - 60	30

Table 5: Problem Sizes

The computational results are tabulated in Table 6. These results are obtained using the parameter values determined by the experiments described in the last section. However, the FACOPT system allows the user to use different parameter values.

For the 10 department problems the SA took 5 CPU seconds on average to reach the final solution while the corresponding value for the GA was 13 CPU seconds. For the 20 department problems, the SA took an average of 1230 CPU seconds while the

GA took an average of 1460 CPU seconds. For the 30 department problems, the SA took 3360 CPU seconds while the GA took 6460 CPU seconds. Thus from a computational viewpoint SA was more economical. But since facility layout is a design issue where the solution is not re-computed often, the solution quality is more important. From this viewpoint, GA did better, as shown in Table 6.

From Table 6 it can be observed that the GA algorithm performs better than the SA for every problem. Further, it can be observed that the difference between the two algorithms was as low as 0.1% (problem 31) and as high as 25.8% (problem 29). Thus, as in the Tam_30 test problem, the wide variance indicates that it may be difficult to predict whether GA will always do better than SA since this also depends on the selection of parameters as explained earlier while discussing the Tam 30 result. Thus in our software, both algorithms are included as this provides more options for the decision maker. The user may employ any one of these algorithms to generate a layout. Ten layouts may be kept for further user examination.

P-No	GA	SA	P-No	GA	SA
1	4150.12	4201.45	31	4493.59	4497.70
2	3158.82	3305.54	32	5684.89	5809.46
3	3683.63	3842.31	33	4826.32	4937.33
4	4269.05	4327.84	34	4487.43	4504.32
5	4708.32	4957.72	35	4661.97	5092.51
6	19828.69	22104.24	36	17465.01	20193.86
7	15800.95	18259.73	37	18175.31	20803.27
8	17187.54	20535.32	38	17172.77	18732.07
9	18626.38	21070.31	39	18708.79	21675.81
10	17970.19	21313.79	40	18806.7	21855.57
11	44724.58	51369.36	41	47924.74	55020.7
12	36471.04	45028.93	42	45100.47	51053.63
13	43435.84	49722.27	43	49502.04	55825.95
14	43927.72	52394.14	44	47399.37	52196.54
15	46529.55	50925.1	45	45588.01	53117.86
16	5885.17	6292.84	46	5319.50	5482.38
17	4667.71	4682.58	47	5824.70	6025.85
18	4892.28	5050.67	48	5085.11	5338.88
19	4222.84	4550.42	49	5602.67	5746.99
20	4949.51	5052.91	50	4813.97	4978.14
21	17217.12	18780.97	51	21258.04	23809.86
22	19976.92	22808.87	52	20857.52	23197.31
23	20173.4	22575.49	53	19299.85	21611.83
24	21601.97	24787.97	54	20111.66	22428.83
25	18944.4	20096.39	55	20704	22627.26
26	38719.82	45269.24	56	50641.78	54747.39
27	41547.04	45647.26	57	45592.81	50103.29
28	38042.81	44953.6	58	48050.14	54523.93
29	45590.98	57341.05	59	46072.62	50284.91
30	39823.41	46267.34	60	50240.57	56295.52

Table 6: Modified literature problems

The FACOPT Software

In this section, the features of the FACOPT software is explained. The system was developed in Visual Basic and runs under Windows.

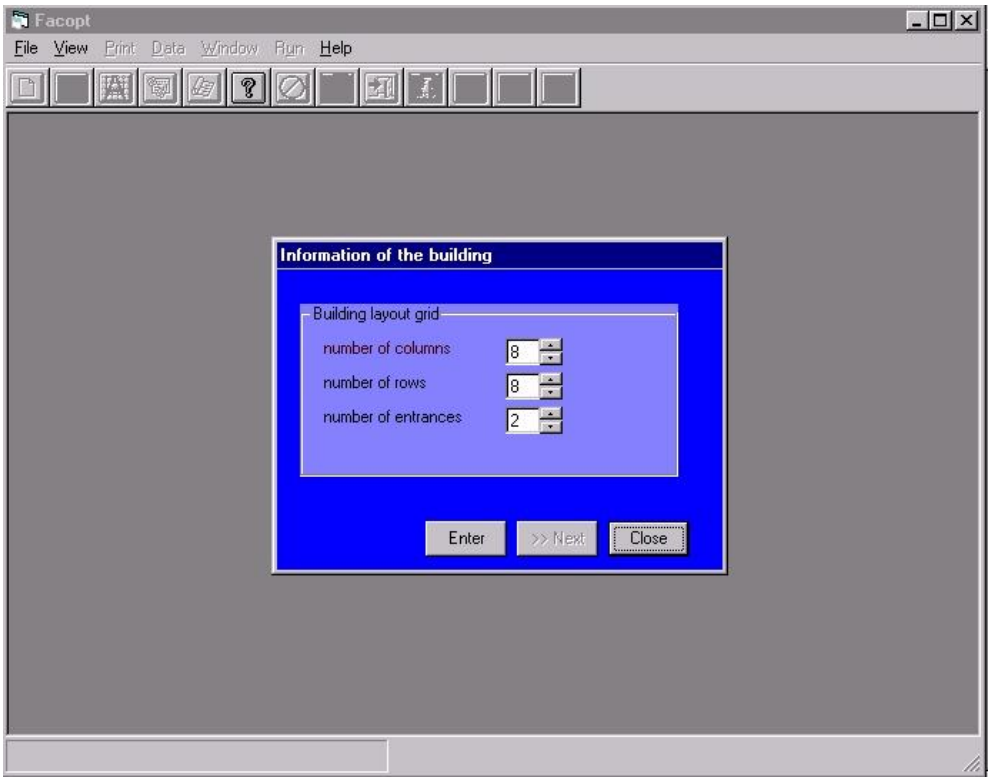


Figure 9: Numbers of columns, rows, and entrances

The user may retrieve a previously created work file or create a new one. In this example, the user is required to enter the number of rows and columns, and the number of entrances (Figure 9). In Figure 10, the user enters department names and sizes and indicates whether a department is fixed. The data entered for a department can be changed by double clicking the department. All entered data will be lost if a reset button is chosen.

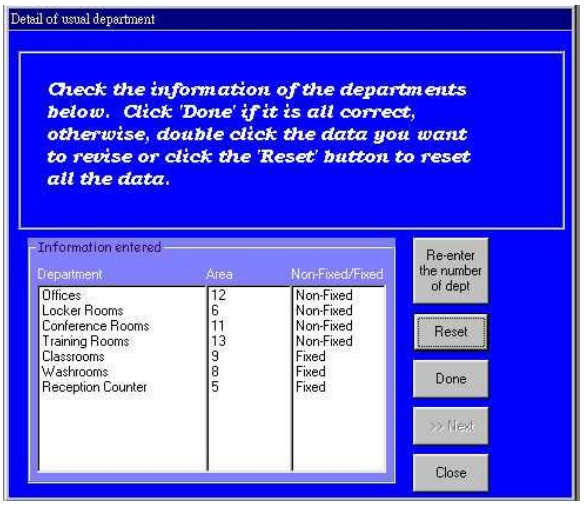
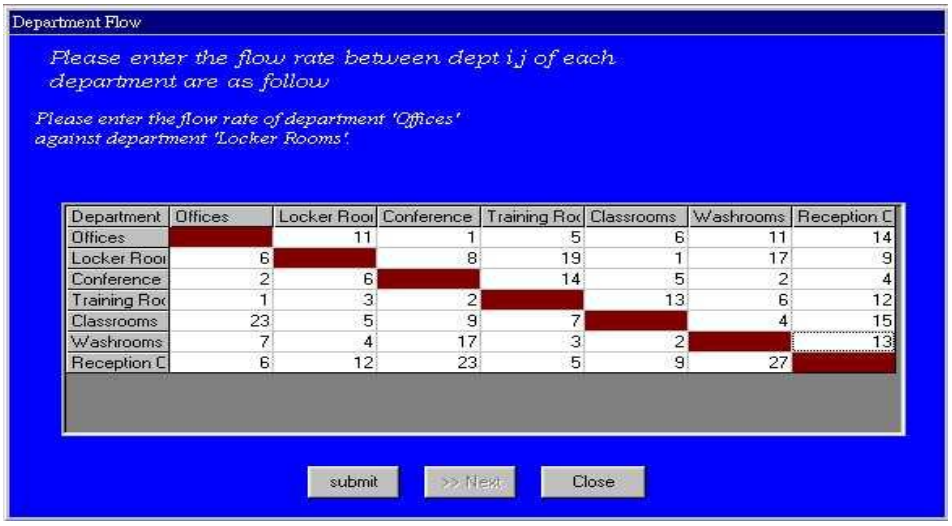


Figure 10: Department names and sizes, and specifying fixed departments

The material flow and unit cost between pairs of departments and those between a department and an entrance may be entered as shown in Figure 11. A SFC can be manually drawn or automatically generated as shown in Figure 12. The user can then select SA or GA as the algorithm. Default parameter values are used. However, the user can change the parameter values (Figure 13).



Department	Offices	Locker Roo	Conference	Training Roo	Classrooms	Washrooms	Reception C
Offices		11	1	5	6	11	14
Locker Roo	6		8	19	1	17	9
Conference	2	6		14	5	2	4
Training Roo	1	3	2		13	6	12
Classrooms	23	5	9	7		4	15
Washrooms	7	4	17	3	2		13
Reception C	6	12	23	5	9	27	

Figure 11: Entering the material flows

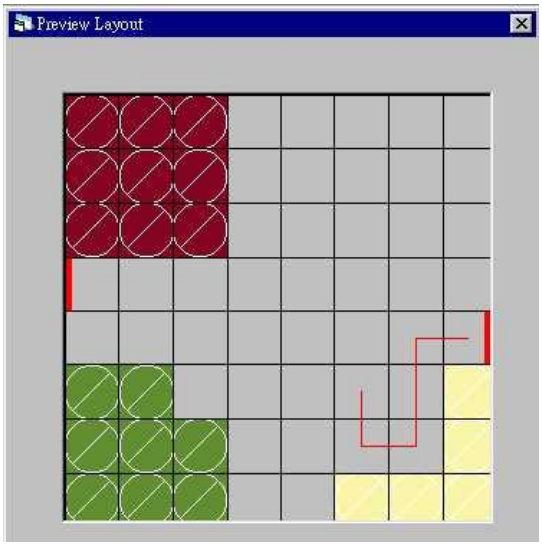


Figure 12: Specifying departments and entrances, and drawing a SFC

Figure 13: Changing the parameter values

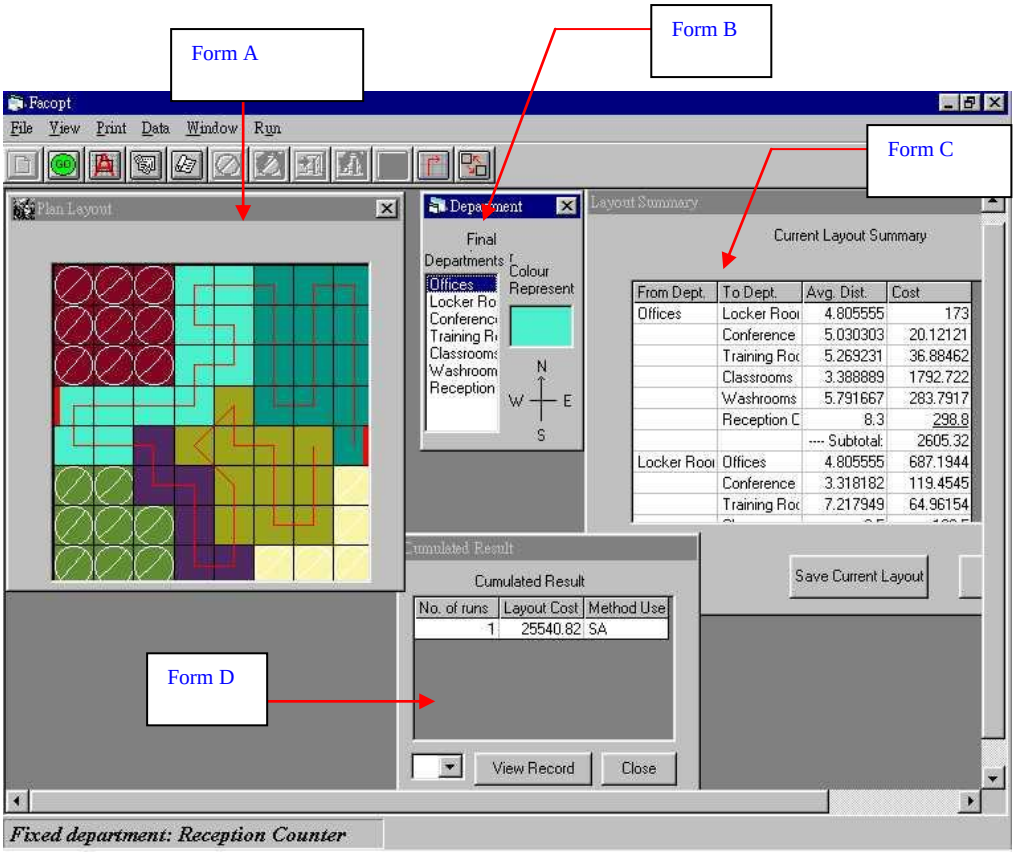


Figure 14: The resulting layout

In Figure 14, Form A is the resulting layout in which each department is represented by a number of square units of the same color. Form B is the department reference. If the user clicks on a department, the color of the department will be shown. Form C shows the average distance and the cost of the material flows between departments. Form D keeps different layout solutions. Ten candidate solutions may be saved.

Conclusion

In this paper, a flexible and user-friendly approach using Visual Basic under Windows to solve the static facility layout problem was proposed. Two effective algorithms can be used to generate heuristic layout solutions. The software was also described. We believe that the proposed system will be useful to layout planners as it is effective and easy to use. Further improvements could include the addition to the software of more solution algorithms such as tabu-search and neural networks for increased flexibility, and enhancements to the software itself.

Acknowledgements:

The authors would like to acknowledge the financial support of the Natural Sciences and Engineering Research Council of Canada (NSERC)

References

1. G.C. Armour and E.S. Buffa, A heuristic algorithm and simulation approach to relative location of facilities, *Management Science*, 9, 294-309, 1963.

2. M.S. Bazaraa, Computerized layout design: a branch and bound approach, *AIIE Transactions*, 7, 432-437, 1975.

3. J.J. Bartholdi and L.K. Platzman, An O(nlogn) planar traveling salesman heuristic based on space filling curves, *Operations Research Letters*, 1, 121-125, 1982.

4. Y.A. Bozer, R.D. Meller, and S.J. Erlebacher, An improvement type layout algorithm for single and multiple floor facilities, *Management Science*, 40, 918-932, 1994.

5. R.W. Eglese, Simulated annealing: a tool for operational research, *European Journal of Operational Research*, 46, 271-281, 1990.

6. J.H. Holland, *Adaptation in Natural and Artificial Systems*, University of Michigan Press, 1975.

7. F.S., Hiller and M.M. Connors, Quadratic assignment problem algorithms and the location of indivisible facilities, *Management Science*, 9, 586-599, 1966.

8. S. Jajodia, I. Minis, G. Harhalakis, and J.M. Proth, CLASS: computerized layout solutions using simulated annealing, *International Journal of Production Research*, 30, 1, 95-108, 1992.

9. R. V. Johnson, Spacecraft for multi-floor layout planning, *Management Science*, 28, 407-417, 1982.

10. L. Kaufman and F. Broeckx, An algorithm for the quadratic assignment problem using Benders' decomposition, *European Journal of Operational Research*, 2, 204-211, 1978.

11. J.G. Kim and Y.D. Kim, A space partitioning method for facility layout problems with shape constraints, *IIE Transactions*, 30, 947-957, 1998.

12. S. Kirkpatrick, C.D. Gelatt, and M.P. Vecchi, Optimization by simulated annealing, *Science*, 220, 671-680, 1983.

13. J.S. Kochhar, B.T. Foster, and S.S. Heragu, Hope: A genetic algorithm for the unequal area facility layout problem, *Computers & Operations Research*, 25, 583-594, 1998.

14. J.S. Kochhar and S.S. Heragu, Multi-hope: a tool for multiple floor layout problems, *International Journal of Production Research*, 36, 12, 3421-3435, 1998.

15. T.C. Koopmans and M. Beckmann, Assignment problems and the location of economic activities, *Econometrica*, 25, 53-76, 1957.

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

16. E.L. Lawler, The quadratic assignment problem, *Management Science*, 9, 586-599, 1963.

17. J. Leung, A new graph theoretic heuristic for facility layout, *Management Science*, 38, 594-605, 1992.

18. R.D. Meller and Y.A. Bozer, A new simulated annealing algorithm for the facility layout problem, *International Journal of Production Research*, 34, 1675-1692, 1996.

19. R.D. Meller and K.Y. Gau, The facility layout problem: recent and emerging trends and perspectives, *Journal of Manufacturing Systems*, 15, 351-366, 1996.

20. S., Sahni and T. Gonzalez, P-complete approximation problem, *Journal of ACM*, 23, 555-565, 1976.

21. J. Skorin-Kapov, Extensions of the tabu-search adaptation to the quadratic assignment problem, *Computers and Operations Research*, 21, 8, 855-865, 1994.

22. K.Y. Tam, Genetic algorithms, function optimization, and facility layout design, *European Journal of Operational Research*, 63, 322-346, 1992.

23. J.A. Tompkins, J.A. White, Y.A. Bozer, E.H. Frazelle, J.M.A. Tanchoco, and J Trevino, *Facility Planning*, 2nd Ed., John Wiley & Son, 1996.

24. K. Tsuchiya, S. Bharitkar, and Y. Takefuji, A neural network approach to facility layout problems, *European Journal of Operational Research*, 89, 556-563, 1996.

Biographies

Jaydeep Balakrishnan is Associate Professor of Operations Management in the Faculty of Management at the University of Calgary, Alberta, Canada. He received his Bachelor of Engineering (Mechanical) degree from Nagpur University in India. His MBA and PhD degrees in Operations Management are from the University of Georgia and Indiana University respectively. He has held visiting appointments at the Chinese University of Hong Kong and the Warsaw School of Economics. He has published in various journals and has made presentations at various international meetings and seminars. He is currently President of the APICS chapter in Calgary. His research interests lie in facility layout and supply chain management.

Chun Hung Cheng graduated from the Chinese University of Hong Kong. He then continued his graduate study at the University of Iowa and obtained his M.Sc. in Computer Science, M.B.A., and Ph.D. in Information Systems. He was a programming analyst in the University of Iowa Hospitals and Clinics while he was a graduate student. In 1994, he returned to Hong Kong and joined the Chinese University of Hong Kong. He is now an associate professor. C.H. is conducting research in Information and Operations Management. His articles have appeared in ACM Transactions on Information Systems, Annals of Operations Research, Computers & Operations Research, Expert Systems, Information and Management, IIE Transactions, IEEE Transactions on Man, Systems, and Cybernetics, Operations Research, and others.

Kam Fai Wong obtained his PhD from Edinburgh University, Scotland, in 1987. After his PhD, he was a researcher at Heriot-Watt University (Scotland), UniSys (Scotland) and ECRC (Germany). At present he is an associate professor in the Department of Systems Engineering and Engineering Management, the Chinese University of Hong Kong (CUHK) and in parallel serves as the associate director of the Centre of Innovation and Technology (CINTEC), CUHK. His research interest centers on Chinese computing and parallel database and information retrieval. He has published over 60 technical papers in these areas in various international journals, conferences and books. He has also organized various conferences and edited books in this field. He is a member of the ACM, CLCS, IEEE-CS and IEE (UK) and a member of the editorial board of the journal on Distributed and Parallel Databases. He was the 1994-96 chairman of the ACM Hong Kong Chapter.