A Hybrid Genetic Algorithm for the Dynamic Plant Layout Problem

Jaydeep Balakrishnan[1]
Chun Hung Cheng[2]
Daniel G Conway[3]
Chun Ming Lau[2]

[1] Operations Management Area
Haskayne School of Business
University of Calgary
Calgary, Alberta T2N 1N4
Canada
Ph: (403) 220 7844 Fax: (403) 284 7902 Email: Jaydeep.Balakrishnan@haskayne.ucalgary.ca

[2] Department of Systems Engineering and Engineering Management
The Chinese University of Hong Kong
Shatin, Hong Kong
PRC
Ph: +852 2609 8322 Fax: +852 2603 5505 Email: chcheng@se.cuhk.edu.hk

[3] Department of Management and Administrative Sciences
Mendoza College of Business
University of Notre Dame
Notre Dame, IN 46556-5646
USA
Ph: (219) 631 4772 Email: dan.g.conway.35@nd.edu

Please direct all correspondence to the first author.

Published in the *International Journal of Production Economics*, 86, 2, 2003, pp 107-120.

A Hybrid Genetic Algorithm for the Dynamic Plant Layout Problem

Abstract

The dynamic plant layout problem (DPLP) deals with the design of multi-period layout plans. Although an optimal solution method based on dynamic programming is available, it is not practical for large DPLPs. It has recently been shown that heuristics based on genetic algorithms can solve large DPLPs. In this research, we extend and improve the use of genetic algorithms by creating a hybrid genetic algorithm. A computational study is carried out to compare the proposed algorithm with the existing genetic algorithms and a recent simulated annealing algorithm. The study shows that the proposed algorithm is effective. Thus it may be useful in solving the larger problems.

Key words: dynamic layout, genetic algorithms, dynamic programming, pair-wise exchange.

A Hybrid Genetic Algorithm for the Dynamic Plant Layout Problem

1. Introduction

This paper investigates the layout problem based on multi-period planning horizons. During this horizon, the material handling flow between the different departments in the layout may change. This necessitates a more sophisticated approach than the static plant layout problem (SPLP) approach. The dynamic plant layout problem (DPLP) extends the SPLP by considering the changes in material handling flow over multiple periods and the costs of rearranging the layout.

The importance of good layout planning can be gauged from the fact that over $250 billion is spent in the U.S. alone on layouts that require planning and re-planning and that 20% to 50% of the total operating expenses within manufacturing can be attributed to material handling (Tompkins et al., 1996).

In an environment where material handling flow does not change over a long time, a static layout analysis would be sufficient. However, in today's market based and dynamic environment, such flow can change quickly necessitating dynamic layout analysis. For example, 75% of Hewlett Packard’s product models are less than three years old and this percentage is increasing (Hammer, 1996). Any change in product mix can result in changes in flow and thus affect layouts. The next section explains the dynamic problem through an example adopted from Rosenblatt (1986).

2. The Dynamic Plant Layout Problem

The dynamic problem involves selecting a static layout for each period and then deciding whether to change to a different layout in the next period. An example of a DPLP is shown in Table A2-1 of Appendix 2. This table shows the material flow for a six department problem over five periods. As seen the relative material flow between departments changes over the planning horizon. For example, in period 1 there is considerable material flow into departments 3 and 4 relative to the other departments. However in period 5, it is departments 1 and 2 that have the major material in-flow. The fixed cost for shifting each department is given at the bottom of Table A2-1.

If the shifting costs are relatively low, we would tend to shift or change the layout configuration more often to suit the changed material handling flow. The reverse is true for high shifting costs where we would avoid relocations to avoid the associated shifting or rearrangement costs. Table A2-2 shows the optimal dynamic solution for problem in Table A2-1 in the form of a layout string where each number denotes a department in the layout. Table A2-3 shows what the actual layout would look like for layout string 246135 in a two row, three column format. The optimal plan is to use layout 246135 during periods 1 and 2. In period 3 departments 5 and 3 switch location. In period 4 departments 6 and 4 switch location and finally in period 5, departments 1 and 6 switch location. In an *n* department, *t* period problem, we would have to implicitly or explicitly evaluate $(n!)^t$ layouts to guarantee the optimal solution. *n!* is the number of possible layout combinations in each period. So, even for a six department problem there are 1.93×10^{14} possibilities. Thus optimal solutions cannot be obtained for practical sized problems and heuristics have to be used. A mathematical formulation for the DPLP is shown below:

$$\text{Min } Z = \sum_{t=1}^{P}\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n}\sum_{l=1}^{n} f_{tik} d_{tjl} X_{tij} X_{tkl} + \sum_{t=2}^{P}\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n}\sum_{l=1}^{n} A_{tijl} Y_{tijl} \qquad (1)$$

s.t.

$$\sum_{i=1}^{n} X_{tij} = 1,\ j = 1,\ldots,n; t = 1,\ldots, P \qquad (2)$$

$$\sum_{j=1}^{n} X_{tij} = 1,\ j = 1,\ldots,n; t = 1,\ldots, P \qquad (3)$$

$$Y_{tijl} = X_{(t-1)ij} \times X_{til},\ \ i, j, l = 1,\ldots, n;\ \ t = 2,\ldots, P \qquad (4)$$

where

P	: Number of periods in the planning horizon
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*
Y_{tijl}	: 0,1 variable for shifting *i* from *j* to *l* in period *t*
X_{tij}	: 0,1 variable for locating department *i* at location *j* in period *t*
A_{tijl}	: Cost of shifting from *j* to *l* in period *t*. ($A_{tijj} = 0$).

Equation (1) is the sum of the material flow and layout rearrangement costs for the planning horizon. Equations (2) and (3) state that each department must be located and each location must be occupied in every period. Equation (4) states that the 0-1 departmental rearrangement variable takes on a value of 1 only if the department shifts its location at the end of a period. The next section reviews the literature in the DPLP.

3. Previous approaches

Rosenblatt uses dynamic programming (DP) to solve the problem with each layout in each period being a state and each period a stage. Since this approach is not practical for large problems, he discusses methods to use DP heuristically. In the heuristic DP method only a few layouts from each period are included. The easy procedure would be to select the layouts in a period randomly. However this is not very effective. A better method is to select some of the better quality layouts in each period to form part of the DP. This gives better solutions though at the cost of additional computation time. Though the DP will give the optimal solution for the layouts included, since not all layouts are included, the solution may not be optimal for the original DPLP. One disadvantage of using only the best layouts in each period is that the best layouts in each period may not be the best from a multi-period perspective. For example, in the problem shown in Table A2-1, the optimal static layouts in periods 1 through 5 are 135642, 142536, 153246, 164253, and 326415 respectively. When compared to the dynamic (multi-period) optimal solution in Table A2-2, it is seen that none of the optimal static layouts form part of the optimal multi-period solution. This is due to the effect of the department shifting costs.

Other procedures have been proposed that perform better than the heuristic DP methods of Rosenblatt. Urban (1993) proposes an approach using a steepest-descent pairwise exchange heuristic similar to CRAFT(Armour and Buffa, 1963). Lacksonen and Enscore (1993) investigate different mathematical programming approaches. Conway and Venkataramanan (1994), and Balakrishnan and Cheng (2000) make use of genetic algorithms (GA) for the DPLP while Kaku and Mazzola (1997) use Tabu Search. Research on dynamic layout when the sizes of departments are unequal have been carried out by among others, Lacksonen (1994), and Montreuil and Venkatadri (1990). Formulations and detailed reviews of these and other approaches to the DPLP can be found in Balakrishnan and Cheng (1998).

More recently, evolutionary approaches (a technique similar to genetic algorithms) have been proposed by Hirabayashi et al. (1999) for a flexible manufacturing system and by Westkamper (2002) for unequal department sizes. Baykasoglu and Gindy (2001) have developed a simulated annealing algorithm (SA) for the DPLP. Using the test problems from Balakrishnan et al., they showed that their algorithm performed better than the GA's of Conway and Venkataramanan, and Balakrishnan and Cheng (2000). The parameter settings in their SA algorithm involve determining the initial temperature (the probability that in the neighbourhood search, an inferior solution will be accepted), the rate at which the temperature decreases (the decrease in the acceptance probability of an inferior solution), and the number of iterations. Evolutionary

One of the weaknesses of the existing GAs in DPLP is that they have not exploited the structure of the DPLP very well. In this paper we investigate some procedures to do this. For example, our crossover operator uses DP to create offspring since DP is an optimisation method for the smaller DPLPs. In addition we use CRAFT in mutation since CRAFT is a well established procedure in layout. Our computational experiments show that this approach provides better quality solutions.

The aim of this research is to develop and test a GA for the DPLP. First, improvements are made to the GA of Conway and Venkataramanan. Their original GA is referred to as the CVGA. Then tests are conducted to determine effective procedures and parameter values for the proposed hybrid GA. Finally this research compares the results of the CVGA, the nested loop GA by Balakrishnan and Cheng (2000), called NLGA, the Baykasoglu and Gindy SA, called the BG, and the proposed hybrid GA using the problem set from Balakrishnan et al. (1992).

4. The proposed hybrid genetic approach

GAs have been widely applied in many different fields such as engineering, physical sciences, social sciences and operations research since its introduction by Holland (1975). They belong to the group of evolutionary programming techniques that resemble the natural selection of genes in living organisms of natural biological systems. Mitchell (1998), and Haupt and Haupt (1998) are recent references for the theory and practice of GA.

A GA based procedure requires that an initial population of feasible solutions be generated. It codes a solution as a finite-length string over some finite alphabet. Each feasible solution may be a parent. The population is also known as the parent pool. The procedure uses information from the objective function to determine the fitness of each potential parent. It obtains good results in different problems through selection, reproduction, crossover, and mutation.

The DPLP is a combinatorial problem for which the optimal solution can be found only for very small problems. A GA based search is an attractive method in dealing with the problem, as GAs are capable of solving large and difficult problems. The proposed GA approach requires many settings and operations as shown in Table 1. In the following sections, the proposed approach is explained.

4.1 Encoding of a solution

The encoding scheme used in the CVGA of Conway and Venkataramanan is used here also. The strings form the population of parents from whom the new generation will be created. Figure 1 shows the scheme. Each static layout (in the six department example, the layout has two rows of three locations each) is represented by a string and the concatenation of the static layout strings forms the dynamic layout string as shown.

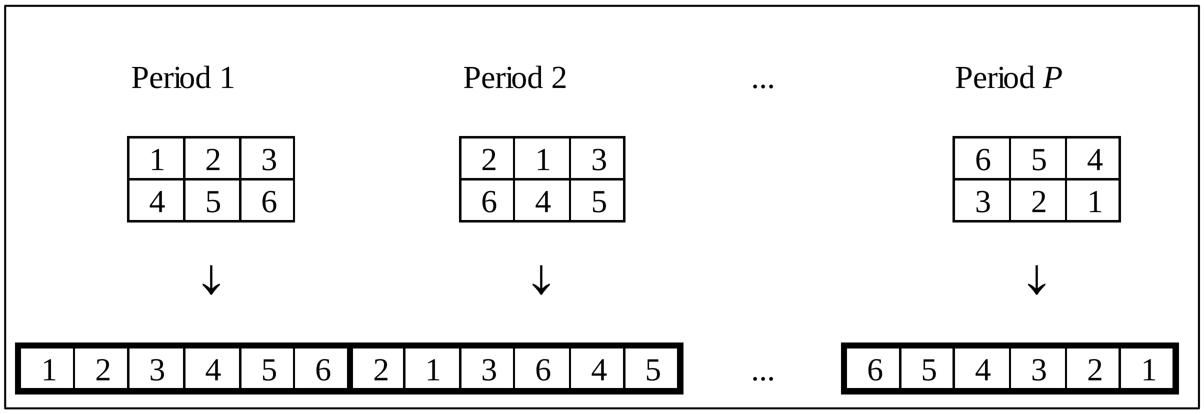

Figure 1: The encoding of a string in dynamic layout

4.2 Fitness function and selection scheme

The fitness function evaluates strings by quantifying the fitness of the strings or solutions. In this case, lower costs imply better fitness The proposed fitness function is identical to the objective function of the DPLP formulation shown in equation (1).

Schemes for selecting parents for crossover or mating are usually random but also biased in the sense that strings with better fitness will have a higher probability of being selected. The tournament selection method was used in the algorithm and is described below.

Tournament selection

The procedure is:

(1) Rank all the parents in the parent pool based on their fitness.

(1) Choose a pre-set value Φ, $(0 < \Phi < 1)$.

(2) Randomly select two parents *P1* and *P2*.

(3) Generate a random number *RN*, $(0 < RN < 1)$.

(4) If $RN < \Phi$ then select the higher ranked parent. Otherwise select the other parent. Thus one parent is selected.

The steps in this process are repeated till *s* parents are generated. This method gave good results during our preliminary tests and was also computationally efficient. Tests with values of 0.6, 0.75 and 0.9 for Φ gave similar results. Thus this method was used in the main experiment with $\Phi = 0.75$.

4.3 Crossover operator

The proposed crossover operator is different from that of the CVGA of Conway and Venkataramanan and the NLGA of Balakrishnan and Cheng. The CVGA uses single point crossover on two strings. First, two strings from the parent pool are randomly selected. Then a point in the two strings is randomly selected and crossover is performed at the chosen point. Although this crossover operator is simple and easy to implement, it may result in infeasible strings that then have to be made feasible. For example consider two strings 123456 and 125643. If the crossover is performed at the sixth position, then there will be two new strings 123453 and 125646. The first string has two occurrences of department 3 and the second string has two occurrences of department 6. Thus further swapping has to be done to make these feasible.

The NLGA uses a point-to-point crossover, which means that the two corresponding departments in every position of the two strings are exchanged to create many child layouts. Some children may be infeasible but some others are guaranteed to be feasible. The NLGA procedure ensures variety by using a nested loop process (where an inner loop uses a traditional GA process and an outer loop generates new parents that replace some the inferior parents in the existing population to create additional variety).

The proposed crossover is quite different from the CVGA and NLGA because it uses an optimisation approach. Unlike the single point or point-to-point crossovers on two strings, it involves many strings and many crossover points. The crossover points are all at the joints of the strings, which is where the layouts of two successive periods meet. This ensures feasibility. The crossover scheme has the following steps:

(1) Select s strings from the parent pool using the tournament selection method.
(2) For a P period problem, cut each string into P equal parts giving $P \times s$ parts in total. Each part represents a feasible static layout. Steps 1 and 2 of the operator are shown in Figure 2 for $P = 3$.
(3) Put all these layouts together and eliminate the duplicated layouts.
(4) Use DP to find the best combination among all the layouts based on fitness. The best multi-period layout plan consisting of one static layout from each period becomes the offspring. Figure 3 illustrates Steps 3 and 4.

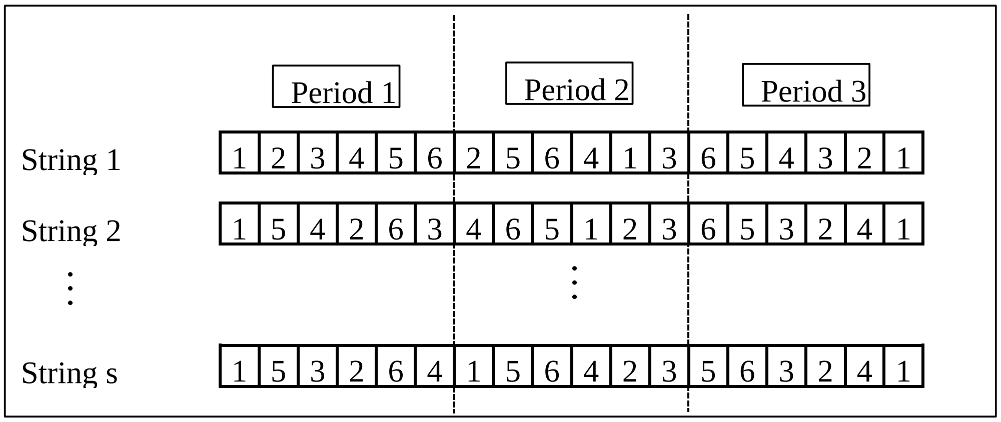

Figure 2: Steps 1 and 2 of the crossover operator

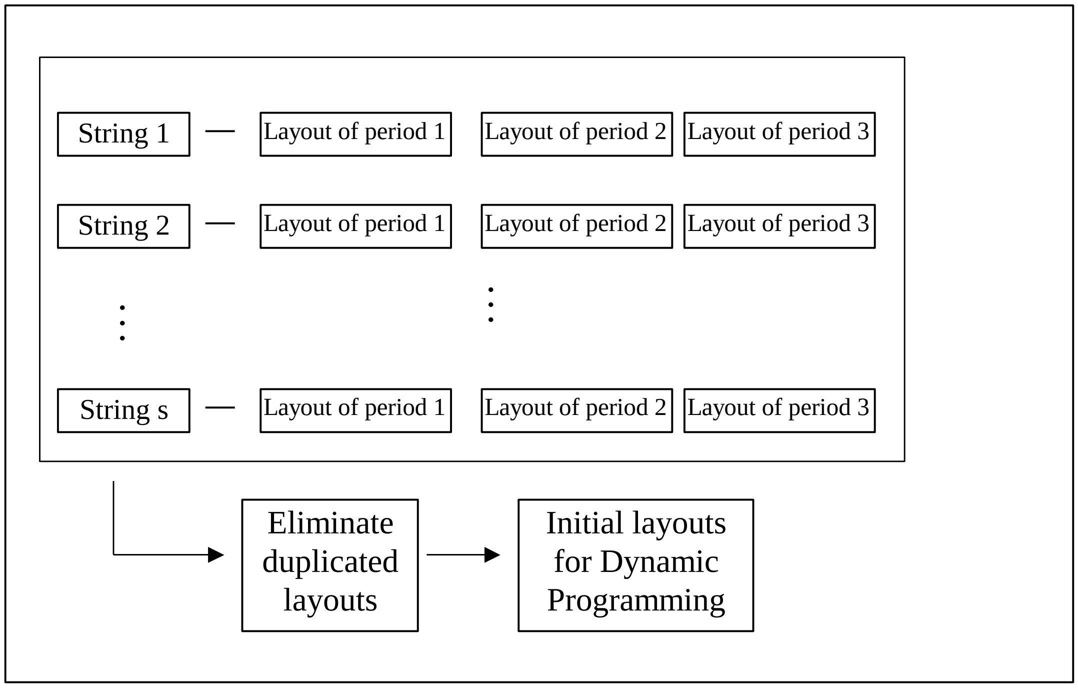

Figure 3: Steps 3 and 4 of the crossover operator

The offspring is obtained using a DP that will have at most $P \times s$ states as shown in Figure 3 which makes it computationally efficient. In this paper the number of states in each stage of the DP is ten as $s = 10$ and the number of stages is either five (for the five period problems) or ten (for the ten period problems) Thus at most there are ten states per stage and ten stages. This means that each stage starting from the penultimate period there will be 10×10 i.e. 100 decisions to evaluate. This has to be repeated nine times for a total of 900 decisions. So the crossover DP can be solved quite easily.

The proposed crossover operator has some important advantages over the operators used in the CVGA and NLGA. First, incorporation of an optimisation approach such as DP and including many strings means that the proposed operator has better holistic view of the problem in determining the offspring. Thus it incorporates the advantages of DP and GA without being computationally prohibitive. In the CVGA and NLGA operators, either pairs of departments position-wise or random sections of two strings are exchanged. Thus they are more local. Second, the proposed operator does not generate infeasible layouts.

4.4 Number of parents considered for crossover

The choice of *s*, the number of parents from the parent pool selected for each crossover operation is an important consideration. The search space depends on *s*. If *s* is large, more parent layouts are considered and the quality of the offspring generated is better. However, the computational requirements also increase.

Since the proposed GA performs crossovers many times (one thousand times), the value of *s* does not need to be very large. The large number of crossovers done compensates for the small value of *s*. The *s* parents chosen each time must be unique. If duplicate parents exist, the number of layouts for DP decreases and the quality of the resulting solution will either deteriorate or remain the same. The search space varies from $(2\times5)^5$ for a 2-parent crossover ($s=2$) in a five period problem to $(20\times10)^{10}$ or 1.02×10^{23} for 20-parent crossover ($s=20$) in a ten period problem. Pilot tests were conducted to determine the *s* value for the main experiment. *s* values of 2, 5, 10 and 20 were used. The *s* parents came from 72 randomly generated parents (the parent pool) and the GA process terminated after 1000 crossovers had been done. The best layout plan was selected as the solution.

Naturally the total cost decreased for larger *s*. However, between $s = 10$ and $s = 20$ though total cost decreased the improvement was only marginal. At the same time the computational time requirement increased significantly. Thus $s = 10$ was selected because it does not require excessive computation time while providing good improvement over smaller *s*.

4.5 Replacement

The method used replaces the weakest parent in the parent pool with the offspring. One disadvantage of this method is that the parent pool may fill up with duplicate parents over generations leading to a lack of variety. As a result the algorithm may converge prematurely. To prevent this, it was decided that the offspring would not replace the weakest parent unless the offspring was unique in the parent pool. This ensured that all the parents in the parent pool were unique. In addition mutation also introduces variety and prevents premature convergence.

4.6 Mutation

Mutation is occasionally applied to the offspring after crossover using a random Bernoulli test. Once the decision has been made to create a mutant, the result is used regardless of whether the mutant affects the quality of the population favourably or unfavourably. Mutation may bring in new static layouts and increase the variety of the parent pool.

The mutation method uses CRAFT, an heuristic improvement algorithm for static layout. The CRAFT procedure involves considering all pair-wise exchanges between each department location with every other department location in the layout. The one among these proposed exchanges that would result in the biggest decrease in the material handling cost in the layout is implemented. This procedure is repeated until no favourable pair-wise exchange can be found. Since it is heuristic algorithm, different initial layouts may result in different final solutions. It is a quick and effective method of redesigning layouts.

In the proposed algorithm, for mutation, the period is randomly chosen and CRAFT is used to generate a new static layout from the existing one. This improved layout (mutant) replaces the original layout. Beside the choice of mutation methods, the mutation rate had to be determined. Grefenstette (1986) recommends a rate of 5% or less. So a value of 5% was used.

4.7 Termination criterion

The termination criterion determines when the GA will stop. In other words, the operations of selection, crossover, reproduction and replacement are repeated until a termination condition is met. In this method, the process is terminated after 1000 generations are performed. Tests showed that 1000 generations were sufficient for convergence. A summary of the choice of parameters and methods is given in Table 1.

Operation	**Procedure**
Encoding	Single string representation
Crossover operator	Dynamic programming
Fitness function	The total cost: sum of flow costs and shifting costs
Selection scheme	Tournament selection
Mutation	Pair-wise exchange heuristic (CRAFT)
Parent pool	Created using Urban's (1993) method
Replacement	Offspring replaces the worst parent if offspring is unique to pool
Termination	Fixed number of generations

Table 1: Summary of the choice of parameters and methods for the proposed GA

5. Computational Study

In this section, the computational study involving the proposed algorithm is discussed. Since it uses DP in crossover, it is called GADP. GADP is compared to the CVGA of Conway and

Venkataramanan and the NLGA of Balakrishnan and Cheng under a variety of situations using the data set from Balakrishnan et al. (1992). This data set consists of eight problems in each of the six situations (6 departments - 5 and 10 periods, 15 departments - 5 and 10 periods, and 30 departments - 5 and 10 periods) for a total of forty eight problems.

5.1 Test problem generation

Twenty four different ten period problems were generated on the computer similar to the one shown in Table A2-1. Eight were six department, ten period problems (layout was a 2 x 3 grid). Another eight consisted of fifteen departments (a 3 x 5 grid) and ten periods. The final eight had thirty departments (a 5 x 6 grid) and ten periods. The five period problems used the first five periods of data from the ten period problems. The material handling flow and the department shifting costs were generated from the uniform distribution. The individual material handling flow (from one department to another) generated were proportionally adjusted so that their sum equalled a pre-set total material flow which was the same for every period in a problem. This was done in order to prevent any period from dominating the others. The generated shifting costs were also proportionally adjusted so that the average department shift cost was 15% of the average department flow cost. Flow dominance was introduced by randomly selecting between 1 and 3 departments from each period and increasing the flow to these departments by a factor of 5 to 7. The shape of the facility and the cost to move a unit distance were held constant over the horizon. The difference between the eight problems was in the ranges of the uniform distribution and the total material flow. For example, in problem 1, the range was between 100 and 200 units of flow while for problem 6 it was between 1000 and 2000.

5.2 Settings

The study consists of the following settings:

(1) Solution techniques: CVGA, NLGA and GADP (R) and GADP (U). The GADP (R) uses random initial strings while GADP (U) generates strings using Urban's pair-wise exchange algorithm (this is explained in the next section). These algorithms were implemented in C programs and were run on DEC Alpha machines.

(2) Sizes of layouts: Three layout sizes of 6, 15 and 30 equal sized departments are used. They represent small, medium and large size problems. Different layout sizes help determine the effectiveness of the algorithms for larger layouts.

(3) Planning horizons: Two different planning horizons of 5 periods and 10 periods are used. This will determine the effectiveness of the algorithms for longer term planning horizons.

The evaluation is based on the total cost, which is the sum of shifting costs and material handling costs. All four methods CVGA, NLGA, GADP (R) and GADP (U) solve all the eight problems for every layout size and planning horizon combination. DP was used to obtain the optimal solutions for the six period problems in order to benchmark the GADP.

5.3 Generating the initial layouts

The GA based search methods need to start from an initial parent pool. Two methods were used to generate the initial parent pool. In the six department problems since all 720 possible static layouts could be generated, these 720 layouts were randomly combined to form the parent strings. In the fifteen and thirty department problems, not all the possible static layouts could be included due to computation time limitations. Thus two methods of generating the strings were investigated.

Random

Randomly generated static layouts are joined to obtain dynamic layout plans for the parent pool. One hundred forty four and seventy two such parents formed the parent pool for the five period and ten period problems respectively. The CVGA, NLGA and GADP (R) always started from the same initial population.

Urban's Pairwise Exchange

One hundred forty four and seventy two parents generated using Urban's procedure formed the parent pool for the five period and ten period problems respectively. Urban's pair-wise exchange procedure is similar to CRAFT. The difference is that shifting costs are included.

Urban's heuristic makes use of forecast windows, *m*, to find different sets of good layout plans for the planning horizon. The forecast window is the number of periods being considered when the pair-wise exchange is performed. It ranges from one to the number of periods *t*. Using an initial layout and pair-wise exchanges, one set of layouts is obtained for the given planning horizon in each forecast window.

For example, when the forecast window is 1, *i.e.*, m=1, in each application of the pair-wise exchange, only material flow from one period are considered. An existing initial layout is used to find the most appropriate layout for period 1 by pair-wise exchanges considering the material flow in period 1 only. Then this newly generated layout for period 1 is used as the initial layout for period 2. Pair-wise exchange is now used to determine a good layout for period 2 by considering the material flow in period 2 only. This process is repeated until the end of the planning horizon. Thus a dynamic layout plan for the entire planning horizon is obtained. The total cost of the layout plan is the sum of the costs of rearrangement and material flow in all the periods.

When the forecast window is equal to 2, the material flows in period 1 and period 2 are added to create a dummy flow matrix and a layout for period 1 is determined using pair-wise exchange. Similarly, flow costs in periods 2 and 3 are combined in determining the layout in period 2. For every m, a layout plan can be obtained. Since each forecast window gives a layout plan, the plan with the lowest cost is selected as the final solution.

Urban's method provides good solutions and is computationally efficient. Our tests also showed that it is an effective method of generating the initial parent pool.

6. Results

Appendix 1 shows the costs achieved by each algorithm in each of the eight problems for every layout size and planning horizon combination. Since Baykasoglu and Gindy had tested their SA using the same problem set, their results are also shown. The computation times were not recorded for every run as the focus of the research was on the design of the layout and solution quality was considered the most important factor. However, in preliminary tests we did monitor the computation time to ensure that the algorithm was able to solve the largest problems in a reasonable time. The thirty department, ten period problems were solved in about 1000 CPU seconds. Thus the proposed algorithm is able to solve practical sized problems in a reasonable amount of time. The SA algorithm of Baykasoglu and Gindy needed an average of over 5000 CPU seconds for the thirty department, ten period problems, (though on another type of machine.).

In the six department problems, the summary results for 5 period and 10 period problems are shown in Tables 2 and 3 respectively. Since all the 720 possible layouts could be considered, DP always provided the optimal solution. In the 5 period problems, the NLGA and GADP are very close in solution quality as both were within 0.1% of optimal on average, while the CVGA does not perform as well. NLGA obtains more optimal solutions than GADP but at the same time, the maximum deviation is also higher. The SA scores in the middle being 0.3% above optimality on average and obtaining only 2 optimal solutions. In the ten period problems, GADP performs the best. On average, the GADP was within 0.1% of optimal while the NLGA SA, and the CVGA did not perform as well on this measure. GADP gave an optimal solution 50% of the time while CVGA, NLGA and SA gave no optimal solutions. The maximum deviations are also the lowest for GADP. Thus it appears that while all the algorithms are adversely affected by the length of the horizon, GADP is the least affected.

	% Deviation from optimal		Number of optimal solutions (out of 8)
	Average	Maximum	
CVGA	0.7	1.6	2
NLGA	0.1	0.5	5
GADP	0.1	0.3	4
SA	0.3	0.8	2

Table 2: Six department and five period summary results

	% Deviation from optimal		Number of optimal solutions (out of 8)
	Average	Maximum	
CVGA	2.1	2.9	0
NLGA	0.4	1.0	0
GADP	0.1	0.6	4
SA	0.6	1.2	0

Table 3: Six department and ten period summary results

For the problems with larger numbers of departments, where computational time limitations prevented the use of the entire static layout pool, optimal solutions were not available. The summary results for the fifteen and thirty department problems are shown in Tables 4 and 5 respectively where the difference between GADP(U) and the other four algorithms are shown.

In the fifteen department five period problems, The GADP (U) performed better than the others. The SA costs were on average 0.4% higher than the GADP (U) and performed better only on one out of the eight problem. In the fifteen department ten period problems, the roles were reversed. The SA costs were 2.8% lower on average than the GADP (U) and performed better in every problem. The GADP (U) in turn dominates the CVGA, NLGA and the GADP (R).

In the thirty department, five period problems (Table 5), the SA performed better than the GADP (U) in five out of the eight problems. SA was on average 0.2% better than the GADP (U). Both dominated the other algorithms. In the thirty department, ten period problems (Table 5) SA dominated all the other algorithms. It was on average 2.5% better than the GADP (U). The GADP(U) in turn dominated the CVGA, NLGA and the GADP (R).

It appears from the results that the GADP (U) is better than the SA for small and medium size problems (all six department and fifteen department, five period problems). For the larger problems the SA seems to be better though the trend is not consistent given the thirty department five period results where GADP (U) was better in three out of the eight problems.

In each of the 32 problems tested under four medium or large problem situations (fifteen and thirty departments, five and ten periods), GADP (U) or SA performed the better than the CVGA, NLGA and the GADP (R). So one of GADP(U) or SA always provided the best solution. Thus a good strategy for practical sized problems might to use both the SA and GADP (U) algorithms to solve any given DPLP since it is not certain which one between the SA and GADP (U) algorithms will give the better result.

It also appears from Tables 4 and 5, based on the average and maximum deviations, that the difference in solution quality between the GADP (U) and CVGA increases for larger department sizes and longer horizons. This did not appear to be true between NLGA and GADP (U) or GADP (R) and GADP (U), as the differences in the performance remained fairly consistent in each of the four situations.

Horizon		Cost increase over GADP (U) solution		
		Mean	Minimum	Maximum
Five Period	CVGA	4.9%	4.2%	6.1%
	NLGA	5.3%	4.6%	6.1%
	GADP (R)	2.2%	1.0%	3.4%
	SA	0.4%	-0.9%[1]	1.4%
Ten period	CVGA	8.4%	6.8%	9.5%
	NLGA	5.9%	5.2%	7.1%
	GADP (R)	2.6%	1.7%	3.7%
	SA	-2.8%[2]	-1.7%[2]	-3.7%[2]

[1] One of eight SA solutions was better than the GADP (U) solution
[2] All eight SA solutions were better than the GADP (U) solutions

Table 4: Fifteen department summary results

Horizon		Cost increase over GADP (U) solution		
		Mean	Minimum	Maximum
Five Period	CVGA	11.8%	9.3%	14.3%
	NLGA	6.9%	5.7%	8.0%
	GADP (R)	3.2%	2.4%	4.3%
	SA	-0.2%[1]	-3.4%	4.5%
Ten period	CVGA	18.6%	16.5%	20.7%
	NLGA	6.1%	5.0%	7.2%
	GADP (R)	3.0%	2.1%	4.1%
	SA	-2.5%[2]	-1.5%[2]	-4.0%[2]

[1] Five of eight SA solutions were better than the GADP (U) solution
[2] All eight SA solutions were better than the GADP (U) solutions

Table 5: Thirty department summary results

The results show that Urban’s method (GADP (U)) is effective in generating the parent pool as it gave a better solution (between 2.2% and 4.3% improvement) in each and every problem than GADP (R), where the parent pool was generated randomly.

It is also seen that the CVGA performed the worst. This may be due to the fact that it uses a simple crossover operator. The crossover may also generate infeasible layouts that have to be made feasible by swaps. Thus it is possible that there may not be enough good children generated by the genetic process. In addition no mutation is used. As mentioned earlier mutation is useful in increasing diversity. Thus the lack of mutation could also explain its poor performance.

7. Conclusion

In this paper the application of a hybrid algorithm for solving the DPLP was investigated. While GA is an attractive solution tool for the DPLP, previous GA’s did not exploit the structure of the DPLP such as using DP in the crossover operator to create offspring, using CRAFT to generate mutations, and using Urban’s method to create an initial population (as explained in Sections 4.3, 4.6 and 5.3 respectively). These techniques have proved to be effective in solving the DPLP in the past. In this paper the GA based hybrid algorithm uses an optimisation based crossover operator and modifies the method of mutation. By conducting some preliminary tests, good settings for the GA parameters were also identified.

Finally the proposed algorithm was compared with the GA's of Conway and Venkataramanan, Balakrishnan and Cheng, and a recent SA algorithm. The results showed the proposed method to be promising as it performed better than the other GAs in every run for the larger problems. Thus it appears that the hybrid approach of combining GA with DP provides better results than using GA alone. This appears to be a result of exploiting the problem structure in both crossover and mutation. The use of Urban's pairwise exchange to generate the initial populations also improves the quality of the solution. It also compares favourably with the SA algorithm, performing better for most small and medium size problems and for some large problems.

Acknowledgement

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

References

1. Armour G.C., and Buffa E.S., 1963. A heuristic algorithm and simulation approach to relative allocation of facilities, *Management Science,* 9 (2), 294-300.

2. Balakrishnan J. and Cheng C.H., 1998. Dynamic layout algorithms: a state-of-the-art survey, *Omega*, 26 (4), 507-521.

3. Balakrishnan J. and Cheng C.H., 2000. Genetic search and the dynamic layout problem: an improved algorithm, *Computers and Operations Research*, 27, 587-593.

4. Balakrishnan J., Jacobs R.F., Venkataramanan M.A., 1992. Solutions for the constrained dynamic facility layout problem, *European Journal of Operations Research*, 57 (2), 280-286.

5. Baykasoglu A., and Gindy N.N.Z., 2001. A simulated annealing algorithm for the dynamic layout problem, *Computers and Operations Research*, 28, 1403-1426.

6. Conway D.G., and Venkataramanan M.A., 1994. Genetic search and the dynamic facility layout problem, *Computers & Operations Research*, 21 (8), 955-960.

7. Grefenstette J.J., 1986. Optimization of control parameters for genetic algorithms, *IEEE Transactions on Systems, Man, and Cybernetics*, 16 (1), 122-128.

8. Hammer M., *Beyond Reengineering*, New York, NY: HarperCollins, 1996, p12.

9. Haupt R.L., and Haupt S.E, *Practical Genetic Algorithms*, New York NY, John Wiley and Sons, 1998.

10. Hirabayashi N., Kita H., and Nagasawa H., 1999. Dynamic facility layout using evolution strategies, *Proceedings of the Second World Manufacturing Congress*, 1999, pp 154-159.

11. Holland J.H., *Adaptation in natural and artificial systems*, Ann Arbor: The University of Michigan Press,1975.

12. Kaku B. and Mazzola J.B., 1997. A tabu-search heuristic for the plant layout problem, *INFORMS Journal on Computing*, 9 (4).

13. Lacksonen T.A., and Enscore E.E., 1993. Quadratic assignment algorithms for the dynamic layout problem, *International Journal of Production Research*, 31 (3), 503-517.

14. Lacksonen T.A., 1994. Static and dynamic layout Problems with varying areas, *Journal of the Operational Research Society,* 45 (1), 59-69.

15. Mitchell, M., *An Introduction to Genetic Algorithms*, Cambridge MA, MIT Press, 1998.

16. Montreuil B., and Venkatadri U., 1990. Strategic interpolative design of dynamic manufacturing system layouts, *Management Science*, 37, 272-286.

17. Rosenblatt M.J., 1986. The dynamics of plant layout, *Management Science*, 32 (1), 86, 76-86.

18. Tompkins J.A, White J.A., Bozer Y.A., Frazelle E.H, Tanchoco, J.M.A, and Trevino J. *Facilities Planning*, Wiley: New York, 1996.

19. Urban T.L., 1993. A heuristic for the dynamic facility layout problem, *IIE Transactions*, 25 (4), 57-63.

20. Westkamper E., (Ed.), 2002. Wandlungsfahige Unternehmensstrukturen fur die variantenreiche Serienproduktion- Report 2000, 2001, 2002: Collaborative Grant 467. Stuttgart; Universitat /Institut fur Industrialle Fertigung und Fabrikbetrieb (IFF).

APPENDIX 1

	Prob 1	Prob 2	Prob 3	Prob 4	Prob 5	Prob 6	Prob 7	Prob 8	Average
Optimal	106419	104834	104320	106399	105628	103985	106439	103771	105224.4
CVGA	108976	105170	104520	106719	105628*	105606	106439*	104485	105942.9
NLGA	106419*	104834*	104320*	106515	105628*	104053	106978	103771*	105314.8
GADP	106419*	104834*	104529	106583	105628*	104315	106447	103771*	105315.8
SA	107249	105170	104800	106515	106282	103985	106447	103771	105527.4

Table A1-1: The total cost of 6 department and 5 period problems

	Prob 1	Prob 2	Prob 3	Prob 4	Prob 5	Prob 6	Prob 7	Prob 8	Average
Optimal	214313	212134	207987	212530	212906	209932	214252	212588	212080
CVGA	218407	215623	211028	217493	215363	215564	220529	216291	216287.3
NLGA	214397	212138	208453	212953	211575	210801	215685	214657	212582.4
GADP	214313	212134	207987	212741	210944	210000	215452	212588	212019.9
SA	215200	214713	208351	213331	213812	211213	215630	214513	213345.4

Table A1-2: The total cost of 6 department and 10 period problems

	Prob 1	**Prob 2**	**Prob 3**	**Prob 4**	**Prob 5**	**Prob 6**	**Prob 7**	**Prob 8**	**Average**
CVGA	504759	514718	516063	508532	515599	509384	512508	514839	512050.3
NLGA	511854	507694	518461	514242	512834	513763	512722	521116	514085.8
GADP (R)	493707	494476	506684	500826	502409	497382	494316	500779	498822.4
GADP (U)	484090	485352	489898	484625	489885	488640	489378	500779	489080.9
SA	484695	486141	496617	490869	491501	491098	491350	496465	491092.0

Table A1-3: The total cost of 15 department and 5 period problems

	Prob 1	**Prob 2**	**Prob 3**	**Prob 4**	**Prob 5**	**Prob 6**	**Prob 7**	**Prob 8**	**Average**
CVGA	1055536	1061940	1073603	1060034	1064692	1066370	1066617	1068216	1064626.0
NLGA	1047596	1037580	1056185	1026789	1033591	1028606	1043823	1048853	1040378.9
GADP (R)	1004806	1006790	1012482	1001795	1005988	1002871	1019645	1010772	1008143.6
GADP (U)	987887	980638	985886	976025	982778	973912	982872	987789	982223.4
SA	950910	947673	968027	950701	948470	948630	965844	956170	954553.1

Table A1-4: The total cost of 15 department and10 period problems

	Prob 1	**Prob 2**	**Prob 3**	**Prob 4**	**Prob 5**	**Prob 6**	**Prob 7**	**Prob 8**	**Average**
CVGA	632737	647585	642295	634626	639693	637620	640482	635776	638851.8
NLGA	611794	611873	611664	611766	604564	606010	607134	620183	610623.5
GADP (R)	603339	589834	592475	586064	580624	587797	588347	590451	589866.4
GADP (U)	578689	572232	578527	572057	559777	566792	567873	575720	571458.4
SA	562405	569251	564464	552684	559596	592515	582409	578549	570234.1

Table A1-5: The total cost of 30 department and 5 period problems

	Prob 1	**Prob 2**	**Prob 3**	**Prob 4**	**Prob 5**	**Prob 6**	**Prob 7**	**Prob 8**	**Average**
CVGA	1362513	1379640	1365024	1367130	1356860	1372513	1382799	1383610	1371261.1
NLGA	1228411	1231978	1231829	1227413	1215256	1221356	1212273	1245423	1226742.4
GADP (R)	1194084	1199001	1197253	1184422	1179673	1178091	1186145	1208436	1190888.1
GADP (U)	1169474	1168878	1166366	1154192	1133561	1145000	1145927	1168657	1156506.9
SA	1122154	1120182	1125346	1120217	1158323	1111344	1128744	1136157	1127808.4

Table A1-6: The total cost of 30 department and 10 period problems

Appendix 2

To -- From	1	2	3	4	5	6
			Period 1			
1	0	63	605	551	116	136
2	63	0	635	941	50	191
3	104	71	0	569	136	55
4	65	193	622	0	77	90
5	162	174	607	591	0	179
6	156	13	667	611	175	0
			Period 2			
1	0	175	804	904	56	176
2	63	0	743	936	45	177
3	168	85	0	918	138	134
4	51	94	962	0	173	39
5	97	104	730	634	0	144
6	95	115	983	597	24	0
			Period 3			
1	0	90	77	553	769	139
2	168	0	114	653	525	185
3	32	35	0	664	898	87
4	27	166	42	0	960	179
5	185	56	44	926	0	104
6	72	128	173	634	687	0
			Period 4			
1	0	112	15	199	665	649
2	153	0	116	173	912	671
3	10	28	0	182	855	542
4	29	69	15	0	552	751
5	198	71	42	24	0	758
6	62	109	170	90	973	0
			Period 5			
1	0	663	23	128	119	50
2	820	0	5	98	141	66
3	822	650	0	137	78	91
4	826	570	149	0	93	151
5	915	515	53	35	0	177
6	614	729	178	10	99	0
			Shifting cost for departments			
	887	964	213	367	289	477

Table A2-1: Material Flow and Shifting Costs

Period	Layout String
1	2 4 6 1 3 5
2	2 4 6 1 3 5
3	2 4 6 1 5 3
4	2 6 4 1 5 3
5	2 1 4 6 5 3

Table A2-2: Optimal solution for the problem in Table A2-1

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

Figure A2-3: The physical layout for layout string 246135