Hybrid Steady State Genetic Algorithm for Layout Problem and the Effect of Using it on Some Types of Crossover

Narjis Mezaal Shati
Department of Computer Science, collage of Science,
University of Al-mustansiryah, Iraq,
email:nareen_manar@yahoo.com

Abstract

This paper investigates a genetic algorithm optimization for the layout problem to show the effect of some type of crossover approaches on the behavior of a steady state genetic algorithm for design optimization. This algorithm is based on suitable techniques including solution encoding, evaluation function definition, and effective crossover operators. It focuses on designing and representing issues of the problem and also shows the effects of some possible types of crossover operators.

A number of test cases using different implementations of genetic algorithms achieve interesting results which illustrate the performance of genetic algorithm present in this paper. So, genetic algorithm (GA) is used in conjunction with local optimizer to find optimal spatial layouts. Where selected the local optimizer is the best choice of refining GA solutions. Different types of crossover used in the modified algorithm gain a better result than other types of crossover chosen.

Key words: Genetic algorithms, Hybrid steady state, Layout problem.

1. Introduction

The space layout planning problem is fundamentally important in many domains from architectural design to very large scale integrated (VLSI) floor-planning, process layouts, facilities layout problems physicists and many other professionals in their attempts to find optimal molecular configurations or arrange components for micro-circuitry design. It can be formalized as a particular case of a combinatorial optimization problem. It presents all the difficulties associated with this class of problems [1].

In layout problems, the major concern is the optimal design or remolding of the layout of an organization [2].

2. Problem Description

An entity consists of an enveloping rectangle non-overlapping basic rectangles (or modules). For every basic rectangle a set of implementations, which have a rectangular shape characterized by a width W, length L and radius R is given. The relative positions of the basic rectangles are specified by the matrix flow [3].

The objective of the layout Problem is to position N objects into an area of length L and width W such that there are no overlaps between the objects. The distances between all pairs of entities are optimized according to a given flow matrix as seen in figure (1).

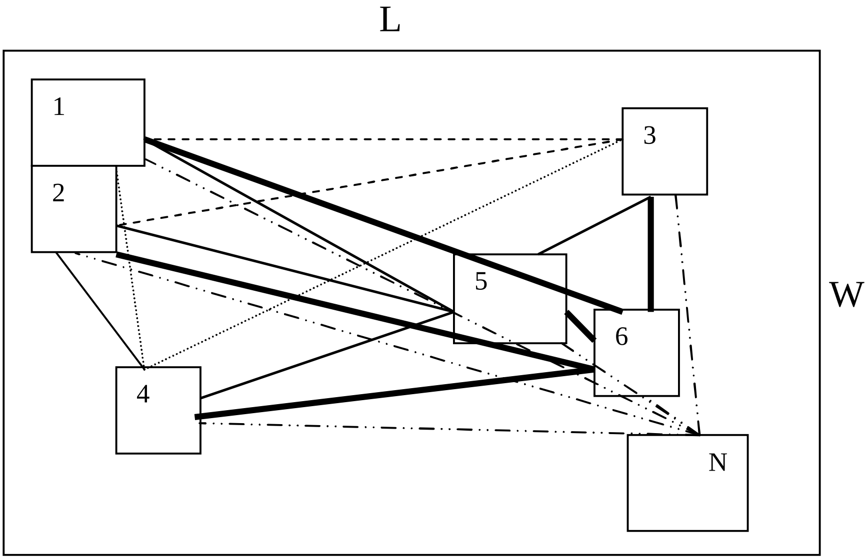


Figure (1) Problem Representation, assuming that the problem is composed of {n} basic rectangles; a list of different implementations is associated with each other through flow matrix represented here in different lines L, W: length, width of area.

The problem can be more succinctly stated given as [3]:

$$\sum_{i}^{n}\sum_{j}^{n} M_{ij}\left\|x-y\right\| \quad \text{----------} \ (1)$$

Where x and y are encoded in a genetic algorithm and indicate the Cartesian coordinates of each individual entity.

A flow matrix defines the flow of connectivity for each pair of entity, it is given as [3].

$$M = [m_{ij}], \ \text{--------} \ (2)$$

Where i,j=1.. n

The objectives to find a non-overlapping arrangement of the entity with minimal flow costs are given as [2].

$$\text{Flow costs} = \sum_{i,j}^{n} d_{ij}.m_{ij} \quad \text{----------------} \ (3)$$

Where d_{ij} is the distance between the centroids of entity layout [4,5].

3. Hybrid Steady State Genetic (HSSG) Algorithm for Designing Problem

Hybrid GAs have been Known as the effective optimization technique for solving the complicated optimization problems.

Hybrid Steady State Genetic (HSSG) algorithm produces one offspring in each generation, then the fitness value of each offspring is measured. After the group of individuals is chosen randomly from population, its efficiency is compared with the efficiency of the offspring. If the best-fit offspring is better than the best-fit chromosome in the population, then the best-fit offspring will be replaced by the population using a replacement strategy, and the updated population is moved to the next generation, otherwise the current population is moved to the next generation without being changed .The details of the algorithm are found in [6].

4. The Proposed Method for Solving the Problem

In this work a genetic algorithm optimization is adopted for the layout problem. The proposed algorithm developed from the hybrid GA is called HSSG.

4.1 Description of (HSSG)

The proposed hybrid genetic algorithm consists of several processes which include representation, genetic parameter, and fitness function, genetic operations (selection technique, crossover operator, and mutation operator). A brief description for the implementation process of HSSG is given as follows:-

```
Step$_1$: Initialization (population);
Step$_2$: Evaluation (population);
Step$_3$: Generation $\leftarrow$ 0;
Step$_4$: While not stops criteria on do
Step$_5$: Selection (population, parent);
          Use: binary tournament
Step$_6$: Crossover (parent, offspring, pc);
          Use: without local optimizer
          Or: with local optimizer
          Or: Pmx
          Or: Cx
          Or: Ox
Step$_7$: Mutation (offspring, pm);
Step$_8$: Separation algorithm;
Step$_9$: Evaluation (offspring);
Step$_{10}$: Replacement (population, offspring);
```

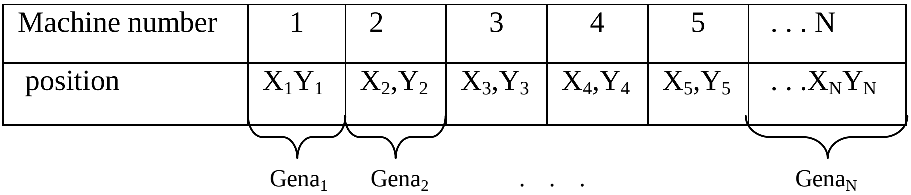
```
    Step$_{11}$: Generation ← generation+1;
    Step$_{12}$: End while;
Step$_{13}$: Extract the best individual according to fitness;
```

4.2 Chromosome Representation

One of the main difficulties in encoding this problem lies in the differences of footprints {center of rectangle that indicates the entity shape} and relationships among them. These differences force each object’s location to be specifically identified in the chromosomes instead of allowing an encoding which could take advantage of similarities among objects. Because the complexity of a problem is related to its chromosome size, and the encoding of the chromosome is related to the number of objects, the complexity increases in proportion to the number of the objects for the particular problem.

Many encodings of the object’s location into the chromosome are tested. These encodings include absolute encodings, where each object has its X and Y position directly encoded on the string (used for representing this problem), and relative encodings where X and Y positions could be determined either relative to another object's position or as an absolute value [7,8].

In this work the Chromosome representation is based on entity numbers, where the location of the gene represents the location of the entity. Figure (2) shows the chromosome representation where {1, 2, ... N} represent the entity number, points (X,Y) represent the position of the entity and the problem representation shown in figure (1) where the lines represent relationships that connect the entities, N represents the number of entities, this work will consider 5, 12, 20, 45, or other number of entity.



Machine number	1	2	3	4	5	. . . N
position	X_1Y_1	X_2,Y_2	X_3,Y_3	X_4,Y_4	X_5,Y_5	. . .X_NY_N

Figure (2) Chromosome Representation

4.3 The fitness function

The fitness function is used to compute the fitness value for the chromosomes of each population such that every one in the population is tested concerning its fitness to find a solution for the coded problem. The fitness value has to correlate in some manner with the suitability of the chromosome by computing a fitness function, and it is used by the selection mechanisms in determining which chromosome will survive and recombine.

In this work the fitness function is obtained from the evaluation function dependent on distance .It takes the distance between each pair of entities and finds the percentage of correctness of that distance relative to the optimal distance indicated by the relationship matrix. The correctness is simply (1- error). This correctness is then multiplied by the relationship flow and added to the total fitness for the given layout. The total fitness, after all relationships, is measured and then divided by the total flow represented in the relationship matrix [2].

4.4 The Genetic Operators

4.4.1 Selection

The binary tournament selection of Whitley's methods in HSSG is employed and is given in [9,10,11].

4.4.2Crossover Operator

The main operators used are with/without local optimizer, and another crossover operator (permutation operator).

- **The Main Operator of Crossover**

1. **Without local optimizer (also called non-optimized):** uses the standard two-point crossover [6,12], when two-point crossover is chosen randomly and the segment between them exchanges here, a problem may arise which is the repetition problem (the two entities occupy the same location). So, using the local optimizer as a kind of greedy algorithm is a good approach in solving this problem.
2. **Local optimizer:** The heuristic crossover (a greedy crossover) constructs an offspring by choosing the best of two parental edges [9,10,13]. This operator constructs an offspring from two parent tours as follows:

Step (1): Pick a random entity (entity) as a starting point for the child's tour.

Step (2): Compare the two edges leaving the starting entity (entity) in the parents and choose the shorter edges.

Step (3): Continue step (2) to extend the partial tour by choosing the shortest of two edges in parents which extend the tour.

Step (4): If the shorter parental edge from step (3) introduces a cycle into the partial tour, then extend the tour by a random edge.

Step (5): Continue until the complete tour is generated.

- **Another Crossover Operators (Permutation Operators)**

The permutation operator can be used for any problems dealing with ordering elements to obtain on optimal solution [6]. In this research work partial mapped crossover (PMX), order crossover (OX), and cycle crossover (CX) are used, see [9,10,14].

4.4.3. Mutation

The general rule of mutation used as follows:

1- Using Parameter Tuning, where the parameter values are set in advance, before the run and are kept constant during the whole execution of the algorithm [14]. This includes Crossover probability (PC), Population size (POP Size), Mutation probability (PM) and Stopping Criterion.

The Mutation probability (PM_1, PM_2) is used. Mutation probability (PM_1) means how often parts of chromosome are mutated. If there is no mutation, offsprings are generated immediately after the crossover (or directly copied) without any change. If the mutation is performed, one or more parts of a chromosome will be changed. If mutation probability is **100%**, the whole chromosome will be changed, if it is **0%**, nothing will be changed [11].

2- The mutation parameter (PM_2) is chosen using trail and error (PM_2=0.5).

3- Generate a random float number named (r_1) between the range [0...1] (if $r_1 < pm_2$) then each gene of the parent has a 0.5 probability of swapping with the adjacent gene of the parent.

One of the practical problems often encountered in GA solution is the problem (overlapping of entities). In order to solve this problem, we use separation algorithms which were considered a good method for solving this kind of problems.

4- If the overlapping problem occurs the separation algorithm will be used as follows [3]:

Step (1):- Sets a priority schedule according to the distance of the entity (entity) from the center of the searues area.

- Those entities close to the center of the area are considered volatile and thus are given lowest priority.
- Those in the corners of the floor are considered fairly stable. This ranking is also similar to measuring the distance that the entity has changed since the last iteration.

Step (2):- Objects are placed according to the priority schedule, with the highest priority being placed first.

- If the object overlaps with another entity, it is moved in the direction in which the two entities overlap and is moved as little as possible to remove the overlap.

Step (3):- This process is iterated until the object is clear of any other entities.

Step (4):-After all objects are placed safely, the algorithm terminates.

5. The Results

In this work the hybrid genetic algorithms are tested on different problems where the number of the entity in (5, 12, 20, and 45 entity) on area has length (L=27), width (W=18), radius (r=2) (i.e. the relation ship between entity and another entity and it takes random), and with open space is distributed. All problems are run by using HSSG.

The entity layout problem is a theoretical problem. In this work it is run with and without local optimizer. Predetermined near global optima was based on the test results received for example we take the 5 and 20 entity.

5.1 The Results of the Test$_1$

The five-entity layout problems is run both with and without local optimizer with the parameter in table (1), and the results are taken from the average of sixteen trials.

Table (١): The parameter that is used in $Test_1$, $Test_2$ Entity Layout Problem

	No. of entity	**size**	**PC**	**PM**	**L**	**W**	**r**
$Test_1$	5	20	0.8	0.1	27	18	2
	20	80	0.8	0.2	27	18	2
$Test_2$	5	60	0.6 - 0.8	0.1	27	18	2
	12	60	0.6 - 0.8	0.1	27	18	2
	20	60	0.6 - 0.8	0.1	27	18	2
	45	60	0.6 - 0.8	0.1	27	18	2

The results of the five-entity layout problem using main operator (without/ with local optimizer) and permutation crossover (PMX, CX, OX) are shown in table (2). These results illustrate that in the five entity layout problem using HSSG with the local optimizer clearly performs much better than the other crossover used in this work, and without the optimizer it indicates the worst results.

Table (2): The result of the $Test_1$ (Five-Entity) Layout Problem in HSSG
using the main operator and permutation crossover

Operator name	**Optimal Goal (successful, total)**	**Best fitness avg.**	**Generation no. avg.**
Local optimize	(16/16)	0.94136	100.55
Without Local optimize	(0/16)	0.63867	1205.2
PMX	(16/16)	0.90016	157.5
CX	(16/16)	0.86453	60
OX	(16/16)	0.88057	67.5

The Twenty-Entity layout problems is run both with and without the local optimizer with Parameter in table (1), and the results are taken from the average of sixteen trials. Table (3) shows the result of the twenty-entity layout problem using main operator (with/ without local optimizer) and permutation crossover (PMX, CX, OX).

Table (3): Results of the $Test_1$ (Twenty-Entity) Layout Problem
using the main operator and permutation crossover

Operator name	**Optimal Goal (successful, total)**	**Best fitness avg.**	**Generation no. avg.**
Local optimize	(16/16)	0.93958	210
Without Local optimize	(0/16)	0.82853	2187.5
PMX	(16/16)	0.88995	107.5
CX	(16/16)	0.90816	87.5
OX	(16/16)	0.89100	43.75

5.2 The Results of the $Test_2$

The result of $Test_2$ is shown in table (4).It shows that there are closely average fitness's except for the without optimizer, and the other operators (pm-cx-ox-loc) are best measured because the average of the best fitness is high and the average number of generation is less.

Table (4): The comparison between the Type of Crossover and the number of entity

No. of Entity	Operator name	Best fitness	Gen
5	**Without Local optimize**	0.70580	1200.5
	Local optimize	0.93430	81.5
	PMX	0.89000	77
	CX	0.91915	80.6
	OX	0.90290	100.5
20	**Without Local optimize**	0.80560	227.8
	Local optimize	0.92050	106.5
	PMX	0.89680	100
	CX	0.90337	70
	OX	0.88910	80
12	**Without Local optimize**	0.70568	3187.5
	Local optimize	0.93856	85
	PMX	0.90735	70.625
	CX	0.91792	113.75
	OX	0.92630	77.5
45	**Without Local optimize**	0.79739	800.6
	Local optimize	0.92141	46.25
	PMX	0.92441	44.9375
	CX	0.91560	50.625
	OX	0.92712	73.125

According to the results given in table (4) the local optimizer operator indicates the best fitness.

5.3 The Results Analysis

This works presents an exhaustive study of the Hybrid Steady State Genetic and the results demonstrate the following:

1. The hybrid search procedure which employs a GA followed by a local optimizer can identify the global optimum for test problems better than without the optimizer; where in the local optimizer solving the problem of repetitions is caused by without optimizer.
2. When using the modified permutation crossover to apply to the problem, it is found that the efficient manner when compared between them illustrates that the choice of local optimizers for refining GA solutions including the modified is the best one.

6. Conclusions

The following conclusions are obtained:

1. The use of the local optimizer crossover operator is better than the other crossover operators.
2. This paper presents an application of genetic algorithm to the problem of finding an optimal layout. The problem is solved as an ordering problem with a major focus on the re-combination of operators and representation issues.
3. Using absolute encoding (entity location) gives a great flexibility in dealing with the problem in spite of the complexity of the problem.
4. Using a fitness function dependent on the distance with the relationship flow gives the best result.
5. The entity repetition problem occurs by using the crossover (with/ without the local optimizer) with (pc). This problem is solved by using the crossover (local optimizer) with 100% crossover without (pc).
6. Using the crossover (permutation operators) gives the best results but with some complexity in the problem.
7. Using parameter tuning with mutation gives the best results.
8. Using separation algorithm solves the over lapping problem that occurs after mutation.

References

[1] C. Papahristodoulou; *"A Pure Binary LP Model to the Facility Layout Problem"*; Department of Economics, Uppsala University; 2001; email: Christos.Papahristodo,ulou@nek.uu.se

[2] E. Goodman; *"An Introduction to GALOPPS (Genetic Algorithm Optimized for Portability and Parallelism system)"*; Tech. Report, Michigan state university; 2003; http://isl.cps.mus.edu/GA.

[3] V. Schnecke, and O. Vornberger, *"Hybrid Genetic Algorithms for Constrained Placement Problems"*; IEEE Transactions on Evolutionary Computation; Vol. 1; No. 4; pp. 266-277; 1998; V. Schnecke is with the Department of Biochemistry, Michigan state University, USA, (Email:Volker@sol.bch.msu.edu); O.Vornberger is with Department of Mathematics/computer Science, University of Osnabruck; Germany, (Email:Oliver @informatik. Uni-osnabrueck.de).

[4] N. Morad; *"Genetic Algorithms Optimization for the Machine Layout Problem"*; School of Industrial Technology, Universities Sains Malaysia 11800, Minden, Penang Malaysia; 1995; Email: nhashima@usm.my

[5] K. Kado; *"An investigation of genetic algorithms for facility layout problems"*; Msc. Thesis; University of Edinburg, UK; 1995.

[6] D. E. Goldberg; *"Genetic Algorithm in search, optimization, and Machine learning"*; Addison-Wesley; 1989.

[7] A. A. Islier; *"A genetic algorithm approach for multiple criteria facility layout design"*; Int'l Journal of production research; Vol. 36; No. 6; PP. 1549-69; 1998.

[8] S. Irani, and H. Huang; *"Custom Design of Facility Layouts for Multi-Product Facilities using Layout Modules"*; IEEE Transactions on Robotics and Automation; Vol. 16; pp. 259-267; 2000.

[9] M. M. Saleiman; *"Study of Per mutative Crossover Behavior on Cryptanalysis of Substitution Cipher and traveling salesman Problems"*; Msc. Thesis; 2002.

[10] J. J. Grefenstette; *"Proceedings of the Second International Conference on Genetic Algorithms"*; Lawrence Erlbaum Associates; Hillsdale, NJ; 1987.

[11] M. Mitchell; "*An Introduction to Genetic Algorithms*"; Printice- Hall; India; 1998.

[12] D. E. Goldberg; *" Genetic Algorithms in Search, Optimization, & Machine Learning"*; Wesley; 1989.

[13] A. R. Ahmad, O. A. Basir, K. Hassanein; *"Improved Placement Algorithm for Layout Optimization"*; Proc. of the 2nd Int'l Industrial Engineering Conf. (IIEC'04); Riyadh, Saudi Arabia; Dec. 2004.

[14] S. E. Uyar, G. C. Eryigit, S. Sariel; "An adaptive Mutation Schema in Genetic Algorithms for Fasting the Convergence to the Optimum"; Istanbul Technical University , Computer Engineering Department; MaslakTR-34469 Istanbul, Turkey;1998. Emails: etaner@cs.itu.edu.tr, gulesn@cs.itu.edu.tr, sariel@cs.itu.edu.tr.

تأثير أنواع التزاوج على سلوك الخوارزمية الجنية الهجينة لحالة الاستقرار لمسالة التوطين

نرجس مزعل شاتي
قسم علوم الحاسبات، كلية العلوم، الجامعة المستنصرية، العراق،
البريد الالكتروني: nareen_manar@yahoo.com

الخلاصة

في بحثنا هذا نوقشت الخوارزميات الجينية الهجينة لمسالة التوطين الأمثل ،لاظهار تأثير بعض طرق التزاوج على سلوك الخوارزمية للتصميم الأمثل. تعتمد الخوارزمية تقنية ملائمة لمسألة التشفير ودالة الصلاحية المعرفة وتأثير بعض معاملات التزاوج عليها. يركز هذا العمل على تصميم وتمثيل المسألة وايضا إظهار تأثير بعض أنواع التزاوج المتاحة.

وقد بينت نتائج هذا البحث من خلال مجموعة من حالات الاختبار التي استخدمت تمثيلات مختلفة للخوارزميات الجينية ان هناك اداء مهم يوضح انجازية الخوارزمية الجنية المعروضة في هذا العمل.