GENETIC SOLUTION FOR BUILDING DESIGN

S.Balasubramaniam C.Kalairaja S.Karthikeyan N.Venkateswaran

Sri Venkateswara College of Engineering, University of Madras, TN, India

Abstract: This paper proposes a novel genetic programming based technique for automatic development
of planning, design and construction of a basic building. The problem uses certain predefined methods or algorithms to get a good and proper building plan. Our design takes care of the main problems such as ventilation, sunlight and air. Our plan is place in such a way that the maximum available space is used. The plan only for a ground floor is only being considered. However this may be extended to generate design for other floors too. In case of extending this plan to industry the algorithm can be modified to include the desired constraints. The inputs are the types of rooms and the area of the building and rooms. The program gives a proper plan placing the dinning , kitchen, bathroom, drawing, bedrooms in the fittest place

Keywords: Genetic algorithms, Building design, Automatic programming, Windows, Plan.

Introduction:

The paper discusses the detail, how Automatic Programming using genetic based techniques can be used to design building plans. The Inputs are given by the user; Using those inputs we have tried to synthesize an optimum output. We are designing the plan for a basic building, but when the number of condition increases the fitness function and hence the optimization work becomes a little tedious. In our paper, we mainly deal with residential type of buildings. Any type of buildings can be constructed by changing the rules and hence the fitness function. Whatever the condition may be, the search procedure remains the same. Since the knowledge about the search domain is insufficient, we use a robust non-linear random, and a multi-modal search method, Genetic algorithm.

In the past, many researchers have used the Genetic algorithm for structural design, (i.e.) designing a part of a building like water pipeline, sewage line, water distribution networks, pressure regulation in water systems, modeling and controlling of congested transportation systems, transportation infrastructure management and
design of roof trusses. We have proposed the genetic algorithm on a new problem of buildingdesign (Environmental design). We have used genetic algorithm to place the rooms of a basic building at appropriate places of the building. This is a new technique we are using to design building plants. We are starting to design with just a basic
residential building. It could be extended to any type of building, but making a few changes. After the basic plan of the building is developed windows are placed by using an another algorithm. By default one window is placed per room to meet the lighting and ventilation conditions.

1) Genetic Algorithm:

The theories of evolution and natural selection were first proposed by Darwin. He observed that, as variations are introduced into a population with each new generation, the less fit individuals tend to die out in competition for food. This is called survival of the fittest principle, and this leads to improvements in the species. The concept of natural selection was used to explain how species adapt to changing environments.

Genetic Algorithms (GAs), invented by John Holland in the 1960s, are based on these principles. They are search algorithms based on the mechanics of natural selection and natural genetics. They combine survival of the fittest among string

structures with a structured yet randomized information exchange.

Like the chromosomes in living organisms, which contain all the information for appearance and behavioral features of a living organism, GA also uses chromosomes (which actually code the solution for the given problem at hand). These constitute the individuals (strings) of the population and are usually binary coded strings (string of 1s and 0s like 110001011), but can also be in other coded forms like real coded strings. It is appropriate to note that those genetic creatures which we just called strings are, in fact, coded solutions to some mathematical model.

For a given problem a suitable chromosome-coding format is chosen and the candidate for solution to the optimization problem, is coded as a chromosome. A population of randomly generated chromosomes is first formed. Reproduction generally involves two parents and the chromosomes of the offspring are generated from portions of chromosomes taken from the parents. Hence the offspring might inherit the characteristics of both the parents. Mutation is changing the genetic content at a particular point in the chromosome. This also introduces new genetic material, which might increase the adaptive ability of the population. When these reproduction and mutation are performed on the initial randomly generated population, the new individuals are formed (next generation) may completely replace the older generation. In every generation, a new set of artificial creatures (strings) is created from the genetic material of the old strings taken from a genetic pool of those strings. And it is important to note that, the candidates for reproduction are not selected randomly. The most fit individual gets to mate more often than a less fit individual. The process of mapping a mathematical model to a sequence of bits is done by a fitness function. The fitness being decided by a mathematical function, which guides the highly competitive individuals in the population towards the solution for the optimization problem. Then the next generation is formed from this new generation, by applying the crossover and mutation operators, and this goes on until some specific condition (like number of generations or the optimized solution) is reached.

After a number of iterations (such generations) the population consists of individuals that are well adapted in terms of the fitness function. Although this setting is reminiscent of a classical function optimization problem genetic algorithms were originally designed to demonstrate the benefit of genetic crossover in an evolutionary scenario, not for function optimization. *It cannot be proven that the individuals of a final generation contain an optimal solution for the objective encoded in the fitness function but it can be shown mathematically that the genetic algorithm optimizes the effort of testing and producing new individuals if their representation permits development of building blocks (also called schemata).*

GAs work with a rich database of points, simultaneously climbing many peaks in parallel, thus reducing the probability of finding and getting stuck on some local peak.

They exploit historical data to focus on new search points, thus exploring most of the search space. The main advantage of using a GA is that, it doesn't get stuck in local minima. Calculus-based methods get stuck in local extremas. Consider a direct method like Hill-Climbing, it finds the local extrema by moving in the direction such as to minimize the slope at the point considered, climbing the hill in the steepest possible direction. But when the objective function is not a single peaked function but multi-peaked one, or if the domain is discontinuous, or if there is not much knowledge about the search space, these methods wont suffice and we require GAs.

The steps involved in a basic genetic algorithm are: -

initialize P(i);// initialize a population of chromosomes, wherein every chromosome is a candidate solution for the given problem although need not be the best

evaluate P(i);//evaluate fitness for all the chromosomes and arrange them in descending order
while <condition> do //while <termination condition like fixed number of generations is not met> do
begin
Select 2 parents from population; //select 2 parent chromosomes from the population using some method like roulette wheel selection
crossover; //crossover these 2 parents to produce offspring (genetic exchange takes place)
evaluate offsprings; //assign fitness values to the offsprings
increase counter **to move towards termination condition;** //move counter toward termination by the given step amount
end
select best chromosome //after all the generations select the best chromosome which will best fit as solution for the given problem and the conditions used in GA to get this

Listing 1

2) Problem Description:

The problem at hand is designing a basic type of building. The user gives the total area, and number of rooms that are to be constructed (like drawing, dining, bedroom, kitchen and bathroom). There are some pre-defined rules such as the place where the rooms have to be placed. The size is allocated to the rooms in the terms of Cartesian co-ordinate system. And the best possible structure of building is to be found using Genetic algorithm.

3) Methodology:

The final building plan consists of rooms and windows of the most optimum fit. The area of the rooms are coded in the binary format. The rooms are coded in terms of coordinate system. The coordinates of the rooms are of fixed length. When a number of conditions are encountered, the fitness function turns to be tedious. This paper, we mainly deal with basic residential type building. The building basically consists of drawing, bedroom, dining, kitchen and Bathroom. It can be applied for any number of rooms. It can be also implemented to design Industrial layouts, buy just changing the conditions of design accordingly. We use genetic algorithm because, once data is accepted from the user, we do not have any idea about the search space. As described earlier, only Genetic algorithm stands a better chance to find the result.

Some basic Conditions are

The plan is represented in 2*n Matrix, in which 2 represents the diagonal co-ordinates of the rooms, n= number of rooms

A chromosome is shown below,

[Drawing:Drawing , Drawing:Drawing…….]
[Bed1:Bed1, Bed1:Bed1]

The 1st line = Drawing room co-ordinate,
2nd line = Bedroom
3rd line = Dining
4th line = kitchen
5th line = Bathroom

If more than 1 room is encountered for each type, the program will work as follows..

Suppose there are two bedrooms that user specifies in input, the second and third line will be taken for bedrooms, and other rooms follow in the following lines.

4) Steps Involved

1) start
2) The chromosomes are randomly generated
3) The chromosomes are evaluated for fitness and the best-fit chromosomes are selected for the crossover
4) The selected population undergoes crossover process
5) The steps 2-4 are run for few specified number of generations
6) After the final generation the fittest chromosome is selected
7) Stop.

5.1) Initial Generation of population:

The chromosomes are initially generated as (2*n) matrix in the order of the preference as shown above. The rules for the generation of chromosomes are taken care of. The total area is given as the input in terms of length and breadth i.e. L and B respectively. The chromosomes are coded in terms of co-ordinates rather than as areas so that the location and the spacing can also be optimized.

5.2 Evaluation: -

The chromosomes so generated are evaluated for their fitness i.e. the rules. The above steps are repeated for a fixed number of times till a best solution is obtained.

5.3 Fitness Function:

This is an essential part of the paper.
The fitness function determines the result of the paper. The fitness is basically decided upon how correctly the rooms are located.

Generally a fitness function decides how the population evolves. By giving proper weight to factors different types of solution for the given design can be evolved. *For example in the design of the fitness function if more emphasizes is laid on accessibility of rooms from one room to another rather than proper placement, then the evolved population will rather will be fit according to the former condition. As an example* we can use the following set of rules while assigning the fitness function: -

The overall plan is not assumed to be a rectangle or a square.

Coolest places are south, east, northeast, southwest.

The total area of the room in terms of its length (L) and its breadth (B) is considered. If found insufficient then the fitness is appropriately reduced.

Using the conditions above we can formulate a table of appropriate fitness on a scale of 10 as given below: -

Rooms And conditions	Fitness
1. BedRoom	
i) In S/E and L/B	10
ii) in place but the length or breadth is not according to dimension (i.e.) approximately greater than 3 foot	4
iii) If the length is not proportional but in the right place	4
iv) If both the above conditions are not true	2
2. Drawing	
i) Accessible to all the rooms and correct dimensions and entrance placed correctly	10
ii) Accessibility is not proper	8
iii) Not proportional	5

Similarly the fitness can be assigned to other rooms also. The ratings for other rooms are as given below: -

3. Dinning Room	
i) Length is proportional and is close to kitchen and drawing	10
ii) Not nearby kitchen	7
iii) Not proportional in length	5
iv) not near to drawing	7

4. Kitchen	
i) If placed in S.E./N.W. and near to dinning room	10
ii) Not near to dinning	7
iii) Lengths are not proportional or area is small	5
iv) Not satisfying any condition.	3

5. Bathroom	
i) If the room is placed west or north and the lengths are proportional	10
ii) Lengths are not proportional	6
iii) The room is not at the proper place	5
iv) None of the above is satisfied	3

Thus each chromosome is allotted fitness according to the above rules. Each and every room is given fitness according to the above set of rules. But however if we need the rooms

To satisfy other conditions they can also be implemented by just giving a value corresponding.

If we need any factor to be more prominent we can scale the factor appropriately.

The final fitness of the function can be got by adding all the fitness assigned to the individual rooms.

This is shown in the example below:-

Consider there are 5 rooms totally.
The rooms are each of one type.
Consider a chromosome that has been generated, Then the individual fitness values assigned to the rooms be given(say) as: -

Bedroom (B)= 10.
Drawing Room (D)= 8.
Dinning Room (G)= 7.
Kitchen (K)= 7.
Bathroom (T)= 6.

The total fitness of the chromosome(genotype) is given as ***F(B,D,G,K,T) = 10 + 8 + 7 +7 + 6***

Therefore the fitness is given by = 38.
In case we want to lay importance on the design of a particular room over other rooms then they can also be incorporated in the fitness function by proper selection of fitness function.

After the ratings are given, the selected chromosomes are taken. The weightage is given according to the preferences. Highest weightage is given to the drawing room and lowest to Bathroom. Hence a desired fitness value is got. Highest of the fitness value is taken as the fittest plan (phenotype).

5.4) Formation of mating pool:

After evaluation of the fitness for all the chromosomes in a particular population, depending on their fitness values, are arranged in decreasing order. During arrangement whenever two circuits

have same value, then the shorter one is given preference to the longer one. A tabulation is made for f(x)/fm where f(x) = fitness of individual chromosome and fm is the mean of all the chromosomes in the population. Then this value is approximated to the closest possible integer. Then final value gives the number of times a particular chromosome will be represented in the mating pool. Then goto CROSSOVER step until the number of generations criteria is met.

5.5) Crossover and Mutation:

The randomly generated chromosomes are evaluated for the fitness and inserted in the mating pool. Such a pool consists of only parent chromosomes, whose number of presence in directly proportional to their fitness.

The matrices are then made to crossover. The crossover takes place in this manner:

The length of drawing room is crossovered with the length of the drawing room only. Only similar chromosomes are crossovered.

Consider Mutation, a chromosome is selected at random and point of mutation in the chromosome is chosen randomly. The application of mutation changes the point 0 to 1 and 1 to 0 randomly. The mutation and crossover probability Pc, Pm are set .

The n after this stage satisfactory result are obtained and the chromosome is then sent for evaluation.

5.6) Example of a chromosome:

A typical chromosome for a 6 room layout(1 dinning,2 bedrooms,1 drawing,1 kitchen , 1 Bathroom) would look like below:

First Co-ordinate Second Co-ordinate
[Drawing:Drawing , Drawing:Drawing……..]
[Bed1:Bed1, Bed1:Bed1]
[Bed1:Bed1, Bed1:Bed1]
[Dinning:Dinning , Dinning:Dinning]
[Kitchen:Kitchen, Kitchen:Kitchen]
[Bathroom:Bathroom , Bathroom:Bathroom]

If more than one room are encountered for each type, then they are sorted as follows: -Suppose there are two bedrooms then the second and third rows are matrix will be taken for bedroom1 and bedroom2 respectively.

Implementation Algorithm:

begin
Initialize population
Evaluate each solution in the population
No of rooms should be initialized// initialising
Total area should be initialized// initialising
step-g:
for generation = 1 **to** (number_of_generations - 1)
call (**formation of the mating pool**)
call (**crossover**)
call (**mutation**)
replace the previous generation by the crossovered population
end for
after last generation select the fittest chromosome
if(fittest chromosome gives the desired outputs)
then
output the Design
else
goto step-g;
end

formation of the mating pool:
{
find the fittest of each chromosome
for I=1 **to** no of chromosomes
x(I)=fitness of the chromosome/mean of the all the fittness
no of counts= approx(x(I))
end for
form the mating pool according to (no of counts)
}

crossover:
{
for crossover= 1 **to** (no of chromosomes/2)

select the pairs of chromosomes from the mating pool randomly and then perform the crossover with the given probability P_c
end for
}

mutation:
{ **for** mutation = 1 **to** no_of_mutations

select a chromosome at random with the probability $_{Pm}$ and mutate a randomly choosen point of the chromosome.
end for

```
}
```

Listing 2

Placing of windows in the plan

Now we move to the other half. After having run through this genetic algorithm, we get the optimum solution i.e. the optimum building plan. Now we have the problem of placing the windows on the plan.

We are able to achieve this by using another genetic algorithm on this already existing plan. We use this genetic algorithm to place the window in the room.

The fitness function is almost same in the sense that it performs the same function in this algorithm also.

Now lets look at the steps involved in this method: -

1) Initial generation of population :

This is pretty much the same excepting that only one room is considered at first. The similar Cartesian co-ordinate style chromosome is used.

The main difference is that each room has its own localized system.

The chromosome is much more smaller in this case.

2) Evaluation :

This stage is similar to the stage in the previous section. Hence no further discussion is needed on this subject.

3) Fitness:

The same sort of fitness function is used.

The value of fitness assigned to the chromosome depends on what parameters we want to optimize the window.

We have decided to adopt the following rules: -

1) The area occupied by the window should not be very large i.e. the area of the window must be very less when compared to of the room.

2) Similarly the area of the window should not be too small.

3) The light let inside the window is also an consideration.

By properly assigning values like the previous section we can get a proper fitness function.

Formation of Mating Pool:

The step was discussed in detail in the previous section. The same procedure is also used in this section too.

The number of times an individual is present is directly proportional to the fitness of the individual.

4) Crossover and Mutation:

The same procedure discussed in the previous section is used here.

Suitable values for the crossover and mutation rate are chosen. They tend to influence the rate at which these occur.

5) Implementation Algorithm:

The implementation algorithm is nearly the same as in the above section. The only difference is that the windows are placed one at a time in one room. This is done so as to reduce the need for computing power.

Conclusion:

We have discussed the method for automatic generation of building plan. The non-deterministic nature of the genetic algorithm assures us that the best possible solution is found out. Yet this type of search and optimization requires some computing power. This can be obtained by the modern processors. We have just shown how automatic programming using genetic algorithms is useful. Still there can be many improvements added to the paper especially in the fitness function that's being used. The interesting point is that a lot of more parameters can be used for optimization of the building plan. We have just given a new area for the application of genetic algorithm. We would like to see much more effort being put to add more parameters for optimization i.e. increase the number of parameters and rules for the fitness function.

References

[1] David Beasley, David R. Bull, Ralph R. Martin. 1993. An Overview of Genetic algorithms: Part 2, Research Topics,*University Computing.*

[2] Sushil J. Louis. 1993. *Genetic Algorithms as a computational tool for design*; PhD.thesis, submitted to, faculty of the University Graduate School,Departmrnt of Computer Science Indiana University.

[3] David E. Goldberg. 1989. Genetic Algorithms in Search, optimization, and machine learning.

Addison-Wesley Longman, Inc.; ISBN: 0-201-47475-1