Genetic Algorithm Approach to Search the Shortest Path for Traveling in Ayeyarwaddy Division

Naing Aung, Khaing Khaing Wai
Computer University, Maubin
naingaung1986@gmail.com

Abstract

Genetic Algorithms (GAs) are adaptive heuristic search algorithms which are premised on the evolutionary ideas of natural selection and gene types. Basically, several random sets of parameters are applied to an algorithm and a fitness value (optimization value) is calculated for each. Based on this fitness values, the best sets are mixed (this is a combination of Selection, Crossover and Mutation) together and new sets are again applied to the algorithm until an optimal parameter(s) are obtained.

This paper presents a genetic algorithm approach to search the shortest path for traveling in Ayeyarwaddy Division. This is based on the analogy of finding the shortest possible distance between towns or cities in a graph or a map. Typically this is represented by a graph with each node representing a city and each edge being a path between two cities. The algorithm has been tested for a road map containing about 127 cities and the experimental results guarantee to provide acceptably good solutions for the given search space.

1. Introduction

Many genetic algorithm models have been introduced by researchers largely working from an experimental perspective. Many of these researchers are application oriented and are typically interested in genetic algorithms as optimization tools.

The shortest path problem is typical in the world of combinatorial systems. This research will attempt to apply a Genetic algorithm to solve this problem based on a real world system. This is based on the analogy of finding the shortest path (i.e. the shortest possible distance) between two towns or cities in a graph or a map with potential connections (assuming that the path distances are always positive). Typically this is represented by a graph with each node representing a city and each edge being a path between two cities. So, applying a genetic algorithm is an interesting idea. This is clearly different from traditional algorithms that try to compare every possibility to find the best solution, which might be a time consuming algorithm for a graph containing a large number of nodes and edges. Applying a genetic algorithm is an interesting idea. This is clearly different from traditional algorithms that try to compare every possibility to find the best solution, which might be a time consuming algorithm containing a large number of towns and paths [4].

2. Related work

The system is designed to make the program to run under a large search space with small space complexity, and to know the ability of the GA to continue to search for better solutions. And, this system reduces can the traveling time, money and fuel for travelers.

This thesis focuses on the development of algorithmic solutions for the best path problem. Finding optimum path has many practical applications within the fields of operations research, logistics, distribution, supply chain management and transportation. In general, best path routing involves finding efficient routes for travelers along transportation routes, in order to minimize route length, service cost, travel time, number of vehicles, etc [4].

3. Theory background

Genetic algorithms are the theoretical background of this system.

3.1 Genetic algorithm

A genetic algorithm is a heuristic technique used to solve optimization problems. Optimization problems attempt to find the best solution(s) for a given problem that has several parameters (goals or resources) with associated constraints. The most basic tools for solving optimization problems are complete enumeration of all possible choices, calculus, and linear optimization techniques using the simplex algorithm such as Lindo or Excel's solver. However, as I will discuss in the next section, these traditional methods break down when the problem gets very large or complicated. Some of the same issues that affect these tools affect genetic algorithms, but for the most part, GAs are far more robust at handling very complex and non-linear problems. For example, a problem involving selecting the best shipping route for a company that needs to make ten shipments may be solved using an 'exhaustive search' technique[3]. The problem could be defined in the computer, which could go through the 3.6 million different combinations of the cities in a reasonable amount of time. However, add many more cities, and you have a problem with what is known as 'combinatorial explosion.' To give you an idea of how unmanageable such a problem can get, review the chart below, which lists the number of permutations associated with up to 25 elements. Such a problem is known as a non-deterministic polynomial problem.

Elements	Permutations
1	1
2	2
3	6
4	24
5	120
6	720
7	5,040
8	40,320
9	362,880
10	3,628,800
11	39,916,800
12	479,001,600
13	6,227,020,800
14	87,178,291,200
15	1,307,674,368,000
16	20,922,789,888,000
17	355,687,428,096,000
18	6,402,373,705,728,000
19	121,645,100,408,832,000
20	2,432,902,008,176,640,000
21	51,090,942,171,709,400,000
22	1,124,000,727,777,610,000,000
23	25,852,016,738,885,000,000,000
24	620,448,401,733,239,000,000,000
25	15,511,210,043,331,000,000,000,000

Figure 1. Permutation of Genetic Algorithm

A genetic algorithm is a heuristic technique that avoids complete enumeration of the solution space by using 'rules of thumb' to find a good solution. What this means is that it does not guarantee the very best solution [4].

3.1.1 Benefits of GAs

The solution time is very predictable, and is not radically affected as the problem gets larger. It can handle non-linear and discontinuous functions equally as well as linear and continuous. You need only to be able to describe a good solution; you do not need to know how to build it. Thus, it does not require heavy use of expert knowledge. It can produce novel results among a set of good solutions. It tends to be compact, containing only the fitness function and a little code to handle the GA functions. It can usually be embedded easily, and are easy to hybridize [3].

3.1.2 Disadvantages of GAs

It is a heuristic; it does not guarantee the optimal solution. Since GA's only drive toward the optimal solution using the fitness function, there is no explanation about how one might logically arrive at the solution [3].

3.2 Components of GAs

The major components of genetic algorithm are:
(i) Initial population
(ii) Evaluate fitness
(iii) Selection
(iv) Crossover
(v) Mutation

(i) Initial population: The size of the population depends on the number of towns in the Map. Many individual solutions are randomly generated to form an initial population. The population size depends on the nature of the problem, but typically contains several hundreds of possible solutions. In the initial population, each individual (chromosome) is the solution for source to destination. Chromosome is represented as an Integer array string. Each town index is a gene of the chromosome and the number of towns in the path may differ from individual to individual .To keep a track of this, each chromosome is given a variable called NUM_TOWNS_PATH to hold the number of towns in the path.

9	3	1	5	6	-	-	-	-	-

Figure 2. An example of a Chromosome. A route from town '9' to '6' is represented through the cities 3, 1, 5.

(ii) Evaluate fitness: Fitness value is calculated for each chromosome according to the given fitness function. The fitness function is the most crucial aspect of any GA. Ideally; the fitness function should be smooth and regular, so that chromosomes with reasonable fitness are to chromosomes with better fitness. And, fitness function may differ according to the nature of problems.
The fitness function is defined as follow:
F(x) = (1/ Actual length of the path) - Disconnected path

(iii) Selection: A proportion of the existing population is selected to breed a new generation. Individual solutions are selected through a fitness based process, where fitter solutions (as measured by fitness function) are typically more likely to be selected. Selection is the procedure for choosing as parents on which to perform crossover in order to create new populations.

The process of selection specifies which chromosomes in the next generation. Popular and well studies selection methods include roulette wheel selection, rank selections and tournament selection.

(iv) Crossover: Crossover operator depends on the chosen encoding and on the problem. The crossover operator produces two new offspring from two parent strings, by copying selected bit from each parent. While there are many different kinds of crossover (such as one point crossover, two point crossover, uniform crossover and arithmetic crossover) one point crossover will be used in this system.

Original

individual 1	1 2 3 6 | 5 8 5 6 7 8 9 10
individual 2	1 2 5 4 | 8 7 3 6 10

After crossover

Offspring 1	1 2 3 6 | 8 7 3 6 10
Offspring 2	1 2 5 4 | 5 8 5 6 7 8 9 10

Figure 3. One point crossover

(v) Mutation: Mutation operator depends on the chosen encoding and on the problem. Mutation operator represents a change in the bit. The mutation operator produces small random changes to the bit string by choosing a single bit at random, then changing its value. Mutation is often performed after crossover has been applied. Mutation operator is intended to prevent falling of all solutions in the population into a local optimum of the solved problem

3.2.1 Algorithm for system

BEGIN

Step(1) Accept source and destination from user;

Step(2) Find all possible routes for source and destination from database;

Step(2) Initializing available routes chromosome;

Step(3) Calculating Fitness;

Step(4) WHILE acceptable result not fulfilled DO

- Select the chromosome to produce new chromosomes;
- Create crossover available routes;
- Eventually mutate some routes;
- Compute fitness of new chromosomes;

Step (5) Report the best final solution;

END

4. Proposed System Design

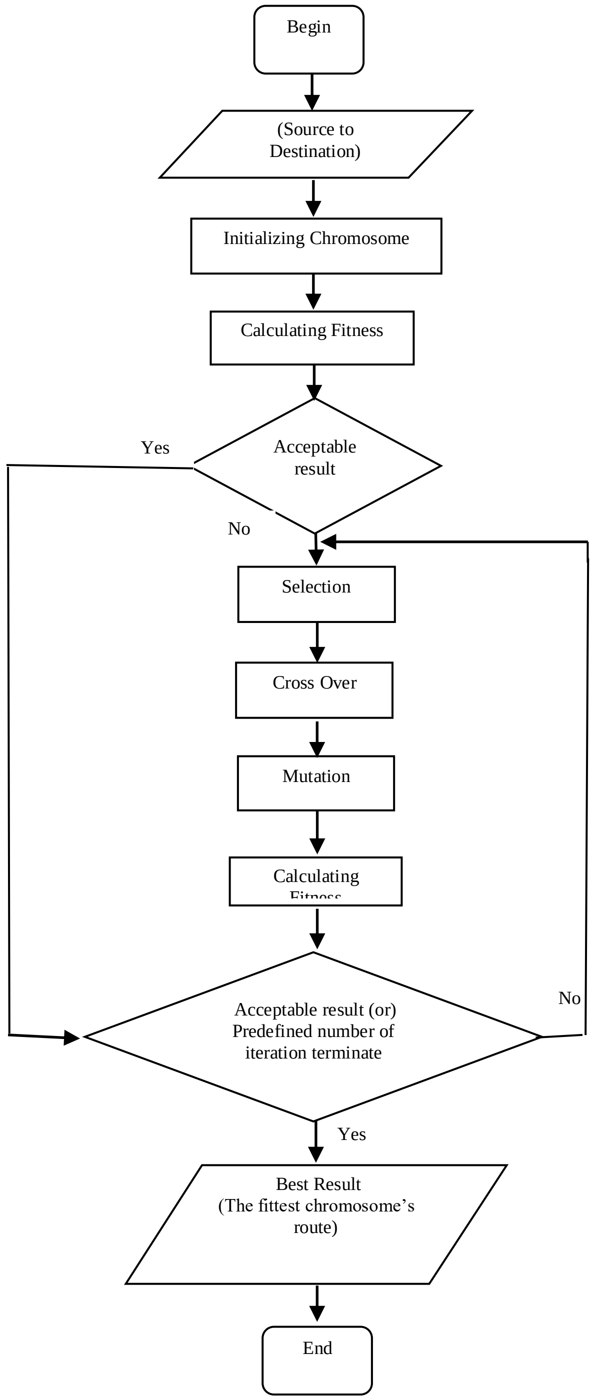

Figure 4 . Flowchart of the system design

5. Implementation of the system

In this system, user chooses source town, destination town in the Ayeyarwaddy Division and search. The system automatically searches the possible routes (initial population) using genetic algorithm and displays to user. If the user does not like this possible ways, then click Crossover. The system will automatically crossover and mutate the possible routes from source town and generates the shortest route of the possible routes.

Danu Phyu-Za Lun-Shan Su-Thaung Gyi-Kyone Pyaw-Yae Kyi-Nga Thaing Chaung-Kyone Tar-I Tha byu-Lay Myet Nar-Ta Lote Taw-In Ga Po-Hin Thada Fitness = 0.0515 Distance=96.5

Danu Phyu-Kyone Ta Nee-Ta Lote Kone-Thaung Gyi-Tha Pyay Hla-Daik Pyet-I Tha byu-Lay Myet Nar-Ta Lote Taw-In Ga Po-Neik Ban-Hin Thada
Fitness = 0.0615 Distance=81.00

Danu Phyu-Kyone Ta Nee-Ta Lote Kone-Thaung Gyi-Tha Pyay Hla-Daik Pyet-Neik Ban-Hin Thada
Fitness = 0.062 Distance=80.3

Danu Phyu-Kyone Ta Nee-Ta Lote Kone-Thaung Gyi-Tha Pyay Hla-Daik Pyet-Neik Ban-In Ga Po-Hin Thada
Fitness = 0.0655 Distance=76.300

Danu Phyu-Kyone Ta Nee-Ta Lote Kone-Thaung Gyi-Shan Su-Neik Ban-I Tha byu-Lay Myet Nar-Ta Lote Taw-Hin Thada
Fitness = 0.071 Distance=70.4

Figure 5. Possible Routes

In figure 5, user chooses the source town (Danu Phyu) and destination town (Hin Thada) and click search. The system populates the possible routes from source to destination and displays the result. In this result, the maximum value is 0.0710 and distance is 70.4 miles.

Danu Phyu-Kyone Ta Nee-Ta Lote Kone-Thaung Gyi-Shan Su-Za Lun-Hin Thada
Fitness=0.074 Distance=67.3

Danu Phyu-Kyone Ta Nee-Ta Lote Kone-Thaung Gyi-Shan Su-Neik Ban-In Ga Po-Hin Thada
Fitness=0.76 Distance=65.7

Danu Phyu-Kyone Ta Nee-Ta Lote Kone-Thaung Gyi-Tha Pyay Hla-Daik Pyet-I Tha byu-Neik Ban-In Ga Po-Hin Thada Fitness=0.076 Distance=65.4

Danu Phyu-Kyone Ta Nee-Ta Lote Kone-Thaung Gyi-Tha Pyay Hla-Daik Pyet-I Tha byu-Lay Myet Nar-Ta Lote Taw-In Ga Po-Hin Thada
Fitness=0.0815 Distance=61

Danu Phyu-Kyone Ta Nee-Ta Lote Kone-Thaung Gyi-Kyone Pyaw-Yae Kyi-Nga Thaing Chaung-Kyone Tar-Tha Pyay Hla-Daik Pyet-Neik Ban-In Ga Po-Hin Thada
Fitness=0.0845 Distance=59.1

Figure 6. Crossover

In figure 6, user click crossover button. The system crossover the possible routes from source and displays the result. In this result, the maximum value is 0.1030 and distance is 48.5 miles.

Danu Phyu-Kyone Ta Nee-Ta Lote Kone-Thaung Gyi-Kyone Pyaw-Yae Kyi-Nga Thaing Chaung-Kyone Tar-Tha Pyay Hla-Daik Pyet-Neik Ban-Hin Thada
Fitness=0.1055 Distance=47.2

Danu Phyu-Kyone Ta Nee-Ta Lote Kone-Thaung Gyi-Shan Su-Neik Ban-Hin Thada
Fitness=0.1105 Distance=45.2
Danu Phyu-Za Lun-Shan Su-Thaung Gyi-Kyone Pyaw-Yae Kyi-Nga Thaing Chaung-Kyone Tar-I Tha byu-Neik Ban-Hin Thada
Fitness=0.1175 Distance=42.4

Danu Phyu-Za Lun-Shan Su-Thaung Gyi-Kyone Pyaw-Yae Kyi-Nga Thaing Chaung-Kyone Tar-I Tha byu-Neik Ban-In Ga Po-Hin Thada
Fitness=0.1200 Distance=41.5

Danu Phyu-Za Lun-Hin Thada Fitness=0.1515 Distance=33.00

Figure 7. The shortest route

In figure 7, the system crossover the possible routes from source and finally displays the shortest route. In this result, the maximum value is 0.1515 and distance is 33 miles.

6. Conclusion

This system presents a genetic algorithmic approach to the shortest path routing problem. Crossover and mutation together provide a search capability that results in improved quality of solution and enhanced rate of convergence. The results are relatively independent of problem types for almost all source–destination pairs. The experimental results show that this algorithm finds more than one possible solution for a given source and destination and this makes it easy to find the next shortest path which exists other than the optimal solution. This solution aims to achieve an increased number of successful and valid convergence using evolutionary computing techniques.

The flexibility of selecting a town can be increased by assigning a fitness value for the routes and town nodes, which are visited more frequently. Then in the next generation the algorithm gives more priority to the cities and routes, which have a higher fitness. The system is intended to extend to search the shortest routes in Myanmar country.

References

[1] Bilal Gonen, "Genetic Algorithm Finding the Shortest Path in Networks" Department of Computer Science and Engineering University of Nevada,
[2] Dr. Hamid Nemati "Genetic Algorithms," Scientific American. July 1992, 66-72.

[3] Peter G. Anderson, Introduction to Genetic Algorithms, Rochester, New York, 2002
[4] Sachith Abeysundara, Baladasan Giritharan. “A Genetic Algorithm Approach to Solve the Shortest Path Problem for Road Maps”, Department of Statistics and Computer Science.
[5] Sam Hsiung, James Matthews, “An Introduction to Genetic Algorithms,” http://www.generation5.org