International Journal of Artificial Intelligence,
Autumn 2008, Volume 1, Number A08
ISSN 0974-0635; Copyright © 2008 by IJAI, ISDER

Finding Shortest Path with Learning Algorithms

Ayoub Bagheri[1], **Mohammad-R. Akbarzadeh-T**[2], and **Mohammad-H Saraee**[3]

[1] Department of Electrical and Computer Engineering
Isfahan University of technology, Isfahan, Iran
ayoub.bagheri@gmail.com

[2] Department of Electrical and Computer Engineering
Ferdowsi University of Mashhad, Mashhad, Iran
akbarzadeh@ieee.org

[3] Departments of Electrical and Computer Engineering
Isfahan University of technology, Isfahan, Iran
saraee@cc.iut.ac.ir

ABSTRACT

This paper presents an approach to the shortest path routing problem that uses one of the most popular learning algorithms. The Genetic Algorithm (GA) is one of the most powerful and successful method in stochastic search and optimization techniques based on the principles of the evolution theory. The crossover operation examines the current solutions in order to find better ones and the mutation operation introduces a new alternative route. The shortest path problem concentrates on finding the path with minimum distance, time or cost from a source node to the goal node. Routing decisions are based on constantly changing predictions of the weights. Finally we arrange some experiments to testify the efficiency of our method. In most of the experiments, the Genetic algorithms found the shortest path in a quick time and had good performance.

Keywords: Crossover, Fitness Function, Genetic Algorithm, Learning algorithm, Mutation, Routing, Shortest Path.

2000 Mathematics Subject Classification: 68Q32, 92D10.

1. Introduction

Routing is one of the important problems in the field of the packet-switched computer network. Network routing algorithms determine routes from source node to destination for communication. In circuit-switching networks as telephone networks, a circuit is allocated between source and destination nodes. Routing algorithms for such networks generate a route on the network status as statistical information, weight of links, and so on. The task of the routing algorithms is to specify how the network should quickly reply to the change of network topology (Liang, Zincir-Heywood, Heywood, 2005).

A routing algorithm can be static or dynamic. In static routing the path used by the sessions of each origin-destination pair is fixed regardless of traffic conditions. Most major packet networks use dynamic routing, where the paths change occasionally in response to congestion. The routing algorithm should change its routes and guide traffic around the point of congestion (Tommiska, Skytta, 2001).

As an important branch of graph and network, the traditional shortest path problem has extensive applications. Finding the shortest path is an important problem in graph theory and has applications in communications, transportation, and electronics problems. This problem plays a central role in the design and analysis of networks.

There are various algorithms for the shortest path problem (Pedrycz,Vasilakos, 2001). Two famous algorithms to calculate the shortest path are the Dijkstra algorithm and Bellman-Ford algorithm, both of which have a time complex of polynomial (Bertsekas, Gallager, 1987). The Genetic Algorithm (GA) based strategy, adapts to the changing network information by rerouting during the course of its execution (Davies, Lingras, 2003).

In this paper, we are interested in to prove that GA can be used to solve a general version of the shortest path problem with the highest possible success rate with a good reasonable time. The rest of the paper is organized as follows. In section 2, the genetic algorithm is reviewed. In section 3 the problem of the shortest path and its history are discussed. We described the proposed GA for the shortest path routing problem in section 3. In Section 4, the proposed algorithm is applied to different networks exhibiting arbitrary link cost, network size. After that, a comparative study of the results follows. The paper concludes with a summary of the results in Section 5.

2. Review of the Genetic Algorithm

Genetic algorithm is a stochastic global search method that looks like the natural process of evolution by using genetic operators. Genetic algorithm has been proved to be efficient in a wide variety of search domains. The genetic algorithm is an optimization and search technique based on the principles of genetics and natural selection. The GA begins, like any other optimization algorithm, by defining the optimization variables, the cost function, and the cost. It ends like other optimization algorithms too, by testing for convergence. In between, however, this algorithm is quite different (Haupt, Haupt, 2004). A path through the components of the GA is shown as a flowchart in Fig. 1.

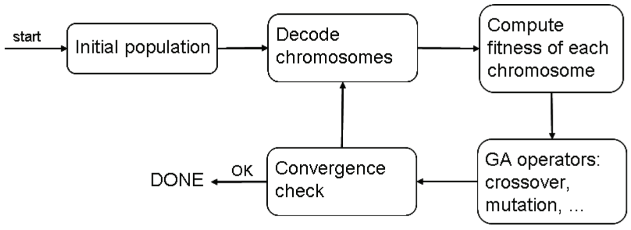

start
Initial population
Decode chromosomes
Compute fitness of each chromosome
GA operators: crossover, mutation, ...
Convergence check
OK
DONE

Fig. 1 routine of Genetic Algorithm

The genetic algorithm (GA) is an optimization and search technique based on the principles of genetics and natural selection. A GA allows a population composed of many chromosomes to evolve under specified selection rules to a state that maximizes the “fitness” (or minimizes the cost function). The method was developed by John Holland (1975). He was the first to try to develop a theoretical basis for GAs through his schema theorem. The work of De Jong (1975) showed the usefulness of the GA for function optimization and made the first concerted effort to find optimized GA parameters (Haupt, Haupt, 2004).

Some of the advantages of a GA include that it

- Optimizes with continuous or discrete variables,

- Does not require derivative information,
- Simultaneously searches from a wide sampling of the cost surface,
- Deals with a large number of variables,
- Is well suited for parallel computers,
- Optimizes variables with extremely complex cost surfaces (they can jump out of a local minimum),
- Provides a list of optimum variables, not just a single solution,
- May encode the variables so that the optimization is done with the encoded variables, and
- Works with numerically generated data, experimental data, or analytical functions.

Genetic operations are crossover and mutation. The crossover operator creates new chromosomes by recombining the genetic material of two chromosomes, deemed the parents. Chromosomes with higher fitness are selected to be parents and ‘‘pass on’’ their genes to the next generation.

Crossovers allow exploitation of successful subspaces of the solution space. The mutation operator randomly alters one or more genes in a chromosome. Mutations add genetic diversity to the population. Through mutation, GAs can search previously unexplored sections of the solution space. Through crossover and mutations, GAs are able to simultaneously explore new subspaces while exploiting successful ones (Davies, Lingras, 2003).

3. Review of the shortest path problem

This problem has many useful applications. The routing protocols used in most of today’s computer networks are based on shortest-path algorithms.

Hence, it has been a subject of extensive research. Dijkstra provided a now well-known algorithm. Floyd relaxed the constraint that all weights must be non-negative. Occasionally it is useful to find two, three or k shortest paths. When the number of nodes is n and the number of edges is m, Eppstein gives an algorithm that finds the k shortest paths (Davies, Lingras, 2003). The approach given in this paper also finds alternate routes as a consequence of its search for the optimal. However, the ranking of these routes is not easily obtained.

4. Proposed GA for shortest path routing problem

There are many heuristic algorithms used to solve complex optimal problems. Genetic algorithms are one of the most powerful and successful methods in stochastic search and optimization techniques based on the principles of the evolution theory. Fig. 2 shows the pseudo code of the genetic algorithm that we employ for the problem of shortest path routing.

```
/*pseudo code of the genetic algorithm*/
repeat
   select two paths Path1 and Path2 from P
   crossover Path1 *Path2 to C1,C2
   for each child path C do
      mutate C with small probabilit y
      replace one solution path in P by C
   end of for
until stopping criterion satisfied
```

Fig. 2 Pseudo code of the genetic algorithm

4.1 Genetic Representation and encoding

In order to apply a genetic algorithm, generally an appropriate encoding of possible solutions in a vector representation is needed.

A directed network G = (V, E, W) with V being the set of nodes (routers), E the set of edges (transmission links), and W the set of weights.

The costs (weights) are specified by the cost matrix W= [Wij], where Wij denotes a cost of transmitting a packet on link (i, j). The cost matrix can be defined as follows in equation (1).

$$\begin{aligned} &W_{ij} = N \\ &(\text{if exists the link from node i to node j, N is positive}) \\ &\mathrm{W}_{ij} = 0\,(\text{otherwise}) \end{aligned} \tag{1}$$

In GAs, each possible solution to a problem is encoded as an individual chromosome. An example of chromosome (routing path) encoding from Source node to Goal node is shown in Fig. 3, Fig. 4. The chromosome is essentially a list of nodes along the constructed path.

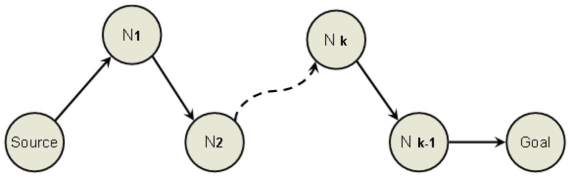
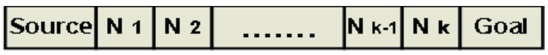


Fig. 3 Example of a routing path



Source	N 1	N 2	....	N k-1	N k	Goal

Fig. 4 Example of encoding scheme of a path (a chromosome)

4.2 Initial population

As in many other applications of the GA algorithms, the population is randomly initialized, creating a population of chromosomes, each made by genes. A person generates an initial set of possible paths based on experience. However, an algorithm can make decisions based on more information. The initial population involves k initial random paths, where k is the size of the population. We initialize a chromosome to a random path but, during evolution, through crossovers and mutations, these paths may change.

4.3 Fitness evaluation

The fitness function interprets the chromosome in terms of physical representation and evaluates its fitness based on traits of being desired in the solution (Chang Ahn, Ramakrishna, 2002). But, the fitness function must accurately measure the quality of the chromosomes in the population. The definition of the fitness function, therefore, is very critical (Chang Ahn, Ramakrishna, 2002). The fitness function in the shortest path problem is obvious because the shortest path computation amounts to finding the minimal cost path. Equation (2) shows the fitness function of the GA procedure.

$$FitnessFunction = -\sum W_{ij} \qquad (2)$$

Where i and j are two nodes and W is weight of links. The fitness function attempts to minimize the cost of path (in other word maximize the Fit). In addition, the fitness function introduces a criterion for selection of chromosomes.

4.4 Crossover

This operator randomly chooses a locus and exchanges the subsequences before and after that locus between two chromosomes to create two offspring (Melanie, 1999). In other words the crossover operation is made by exchanging a portion of genetic materials (genes) between parent chromosomes. It is a probabilistic operation which is controlled by a pre-specified parameter, namely crossover probability. When it is activated, a pair of parent chromosomes is selected randomly for the mating pool, and then replaced by a new pair of offspring generated from parents [].

The crossover operator is the recombination of strings among two randomly selected individuals in the intermediate population to create new offspring for the next generation (Tommiska, Skytta, 2001).

Crossover examines the current solutions in order to find better ones. Physically, crossover in the shortest path problem plays the role of exchanging each partial route of two chosen chromosomes in such a manner that the offspring produced by the crossover represents new routes. The crossover between two parents chosen by the selection gives higher probability of producing offspring (Chang Ahn, Ramakrishna, 2002). Fig. 5 shows an example of the crossover procedure.

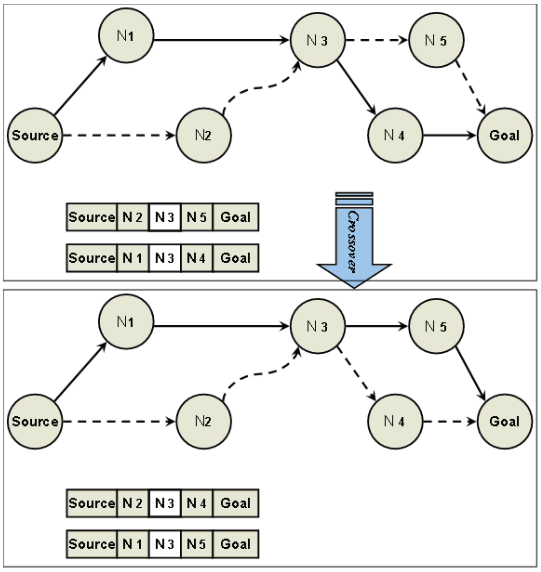

N1
N3
N5
Source
N2
N4
Goal
Source | N 2 | N3 | N 5 | Goal
Source | N 1 | N3 | N4 | Goal
Crossover
N1
N3
N5
Source
N2
N4
Goal
Source | N 2 | N3 | N 4 | Goal
Source | N 1 | N3 | N5 | Goal

Fig. 5 Example of the crossover operation

4.5 Mutation

This operator randomly flips some of the bits in a chromosome. It can introduce traits not in the original population and keeps the GA from converging too fast before sampling the entire cost surface. Mutation points are randomly selected. Increasing the number of mutations increases the algorithm's freedom to search outside the current region of variable space. It also tends to distract the algorithm from converging on a popular solution (Haupt, Haupt, 2004), (Melanie, 1999).

Mutation is the random alteration of one or more genes with a given probability (Xiaoyu Ji, Iwamura, Zhen Shao, 2001). This operator chooses two genes at random and replaces the path between them with a random path. In other words the mutation operator is used to prevent the search process from converging to the local optima prematurely. It diversifies the research process to explore the candidate solution on the solution space in a random way. It is another probabilistic operation which is controlled by another pre-specified parameter, namely mutation probability. The mutation probability is normally very small [].

Fig. 6 shows the Example of the mutation operation.

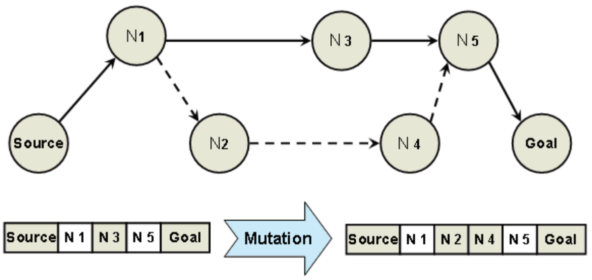


Fig. 6 Example of the mutation operation

5 Implementation, experimentation and results

In order to evaluate the performance of the proposed genetic algorithm for shortest path problem, we designed and implemented a system in C#.Net environment run on a pc with a Pentium IV 2.4GHz CPU and 256MB RAM to solve randomly generated problems.

In this section, we present the results for different network configurations with the randomly generated network topologies. The graph generator first distributes N nodes with N*N cost matrix. The cost between any two nodes n_i and n_j is set by the random probability from 0 to 100.

The simulation studies involve the weighted network topology (with 8 nodes) depicted in Fig. 6. The bold lines show an optimal path. With a view to compare proposed algorithm on the basis of performance with respect to the Dijkstra Algorithm, as shown in the Fig. 6 it is seen that the proposed GA exhibits the shortest path as the result of Dijkstra Algorithm. Fig. 6 (b) shows the shortest path (optimal path) in the graph using Dijkstra algorithm and the Fig. 6 (a) proves the optimality of the proposed GA approach.

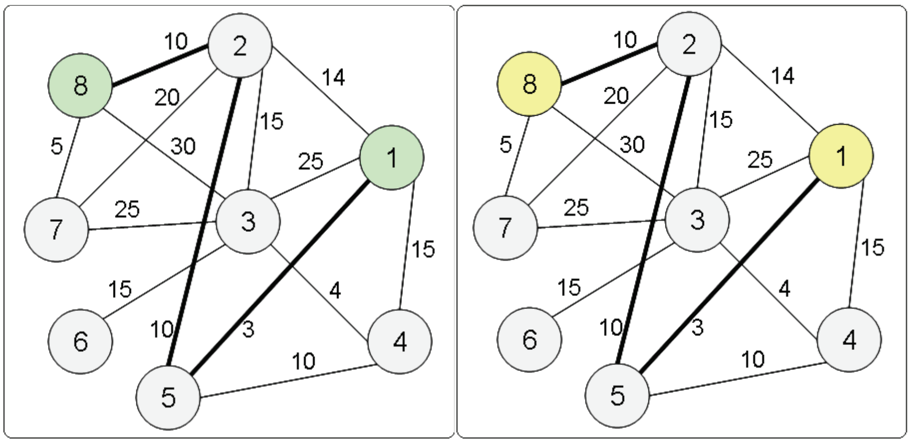


Fig. 6 Example of finding the shortest path in a weighted network topology with 8 nodes. (a) result of the proposed algorithm (total path cost: 23). (b) the optimal path of the graph founded with Dijkstra Algorithm in bold line (total path cost: 23).

The results for a directed complete graph are discussed for illustration. The graph has 100 nodes. The cost matrix of graph is 100*100. Table 1 shows the values of the parameters of GA that used in test 1.

Table 1 the parameters of GA that used in test 1

parameters	Value
population size	10
number of generations	10
crossover ratio	1
mutation ratio	0.01

In most of the test cases, the GAs found the shortest path very fast. In this test the GAs found the shortest path in second generation. Even when the best path is not found, the path returned is still reasonably good. Having these alternate paths can be advantageous in some situations. Practical tests on real traffic data are needed to show if the GA is useful for networks.

Fig. 6 shows results of test 1.

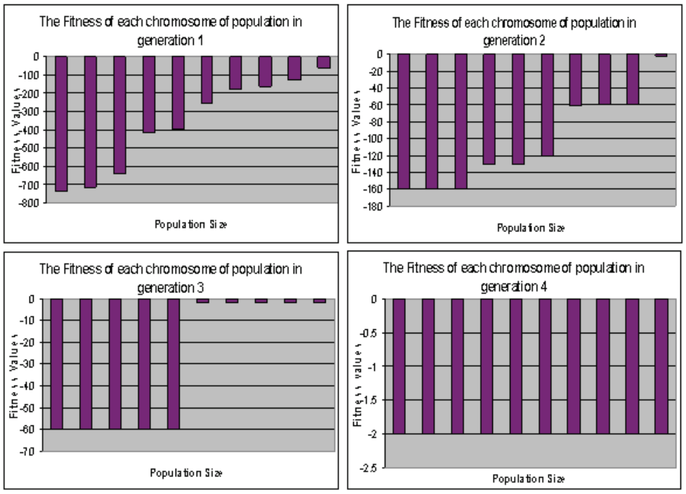


Fig. 6 Result of test 1

A Dijkstra's algorithm was applied to the network and on average the GA outperformed the Dijkstra with regard to time. To justify this the GA and Dijkstra's algorithm were applied to networks with 20 and 80 nodes. The parameters of GA were set as shown in Table 2.

Table 2 Parameters of GA for comparison GA and Dijkstra's algorithm

Number of generation	Population size	Crossover ratio	Mutation ratio
10	20	1	0.01

In each case the algorithms find the shortest path, but a shortest path has to be computed within a very short time in order to support time-constrained services such as voice-, and video-conferencing. The genetic algorithm finds the shortest path faster than the Dijkstra's algorithm. Table 3 presents the results of this test.

Table 3 Comparison of GA and Dijkstra's algorithm

Approach	Nodes	Size of cost matrix	Elapsed time
GA	20	20*20	260 ms
	80	80*80	390 ms
Dijkstra	20	20*20	1420 ms
	80	80*80	4000 ms

The crossover operator is the recombination of strings among two randomly selected individuals in the intermediate population to create new offspring for the next generation. The crossover operation can make convergence of GA faster. The mutation operation randomly flips some of the bits in a chromosome. In all the experiments, the mutation probability is set to a small probability. If the mutation probability had slightly high probability may insert delay to the convergence of GA.

For comparison to other methods, three approach are selected and will be discussed which are (Chang Ahn, Ramakrishna, 2002), (Munemoto, Takai, Sato, 1998), (Inagaki, Haseyama, Kitajima, 1999). Inagaki (Inagaki, Haseyama, Kitajima, 1999) proposed an algorithm that employs fixed (deterministic) length chromosomes. Munemoto's algorithm (Munemoto, Takai, Sato, 1998) is practically feasible in a wired or wireless environment. It employs variable-length chromosomes for encoding the problem. Chang's algorithm (Chang Ahn, Ramakrishna, 2002) employs variable-length chromosomes and their genes have been used for encoding the problem. The measure for performance comparison is the route failure ratio. The route failure ratio is the inverse of route optimality. It is asymptotically the probability that the computed route is not optimal, because it is the relative frequency of route failure (Chang Ahn, Ramakrishna, 2002). The Chang's algorithm (Chang Ahn, Ramakrishna, 2002) always found the optimal shortest path. Where the path computed by the Munetomo's (Munemoto, Takai, Sato, 1998) and Inagaki's (Inagaki, Haseyama, Kitajima, 1999) algorithms, on the other hand, settle for a suboptimal path.

The results are collected in table 4. It means that the proposed algorithm retains its robustness against changing network structure.

Table 4 performance comparison

Algorithms	Inagaki's algorithm	Munemoto's algorithm	Chang's algorithm	Proposed algorithm
Route Failure Ratio (average)	0.4195	0.2959	0.1712	0.1667

The proposed GA achieves a 0.1667 route failure ratio (83.33% route optimality) with a crossover ratio equal to 1 and mutation ratio equal to 0.01.

6. Conclusions

This paper presented a genetic algorithm for solving the shortest path routing problem. A crossover operator is used to interchange the elements of two strings, while mutation operator tries to lead the search out of local optima. The results show that the Shortest-path Routing Based on Genetic Algorithm was a fast algorithm. GA can solve the simpler problem in most cases with minimum failure ratio. The performance of the genetic algorithm could be increased by altering the parameters or optimizing the GAs.

References:

C. Davies, P. Lingras, 2003, Genetic algorithms for rerouting shortest paths in dynamic and stochastic networks, European Journal of Operational Research 144 27–38.

S. Liang, A.N. Zincir-Heywood, M.I. Heywood, 2005, Adding more intelligence to the network routing problem: AntNet and Ga-agents. Applied Soft Computing.

Wiltord Pedrycz, Athanasios Vasilakos, 2001, Computational intelligence in telecommunications networks, by CRC Press LLC.

Dimitri Bertsekas, Robert Gallager, 1987, DATA NETWORKS, Prentice-Hall Inc.

Chang Wook Ahn, Student Member, IEEE, and R. S. Ramakrishna, Senior Member, IEEE, 2002, A Genetic Algorithm for Shortest Path Routing Problem and the Sizing of Populations. IEEE Transactions on evolutionary computation, Vol. 6, No. 6.

Randy L. Haupt, Sue Ellen Haupt, 2004, Practical genetic algorithms, second edition, A John Wiley Inc publication.

Xiaoyu Ji, Kakuzo Iwamura, Zhen Shao, 2001, New models for shortest path problem with fuzzy arc lengths.

Matti Tommiska, Jorma Skytta, 2001, Dijkstra's shortest path routing algorithm in reconfigurable hardware, springer-verlag. pages 653-657.

Mitchell Melanie, 1999, An introduction to genetic algorithms, A Bradford Book the MIT Press, Fifth printing.

M. Munemoto, Y. Takai, Y. Sato, 1998, A migration scheme for the genetic adaptive routing algorithm, in proc. IEEE Int. Conf. Systems.

J. Inagaki, M. Haseyama, H. Kitajima, 1999,a genetic algorithm for termining multiple routes and its applications, in Proc. IEEE Int. Symp. Circuits and Systems.