Genetic Algorithms with Elitism-based Immigrants for Dynamic Shortest Path Problem in Mobile Ad Hoc Networks

Hui Cheng, Shengxiang Yang *Member, IEEE*

Abstract—**In recent years, the static shortest path (SP) problem has been well addressed using intelligent optimization techniques, e.g., artificial neural networks (ANNs), genetic algorithms (GAs), particle swarm optimization (PSO), etc. However, with the advancement in wireless communications, more and more mobile wireless networks appear, e.g., mobile ad hoc network (MANET), wireless sensor network (WSN), etc. One of the most important characteristics in mobile wireless networks is the topology dynamics, that is, the network topology changes over time due to energy conservation or node mobility. Therefore, the SP problem turns out to be a dynamic optimization problem (DOP) in MANETs. In this paper, we propose to use elitism-based immigrants GA (EIGA) to solve the dynamic SP problem in MANETs. We consider MANETs as target systems because they represent new generation wireless networks. The experimental results show that the EIGA can quickly adapt to the environmental changes (i.e., the network topology change) and produce good solutions after each change.**

I. Introduction

Mobile ad hoc network [1], [2], [3] is a self-organizing and self-configuring multi-hop wireless network, comprised of a set of mobile hosts (MHs) that can move around freely and cooperate in relaying packets on behalf of one another. MANET supports robust and efficient operation by incorporating routing functionality into MHs. In MANETs, unicast routing establishes a multi-hop forwarding path for two nodes beyond the direct wireless communication range. Routing protocols also maintain connectivity when links on these paths break due to effects such as node movement, battery drainage, radio propagation, or wireless interference. In multi-hop networks, routing is one of the most important issues that has significant impact on the network's performance. So far, there are mainly two types of routing protocols in MANETs, namely, topological routing and geographic routing. In topological routing, mobile nodes utilize topological information to construct routing tables or search routes directly. In geographic routing, each node knows its own position and makes routing decisions based on the destinations position and its local neighbors' positions.

In this paper, we investigate the shortest path routing, which belongs to the topological routing. The shortest path problem concerns with finding the shortest path from a specific source to a specific destination in a given network while minimizing the total cost associated with the path. The SP routing problem involves a classical combinatorial optimization problem arising in many design and planning contexts [4], [5]. There are several search algorithms for the SP problem: the Dijkstra's algorithm, the breadth-first search algorithm and the Bellman-Ford algorithm, etc. All these algorithms have polynomial time complexity. Therefore, they will be effective in fixed infrastructure wireless or wired networks. But, they exhibit unacceptably high computational complexity for real-time communications involving rapidly changing network topologies [5], [6]. Therefore, for the dynamic SP problem in a changing network environment, we need to employ new appropriate approaches.

Since the static SP problem is a combinatorial optimization problem, the dynamic SP problem turns out to be one of the dynamic optimization problems. In recent years, studying EAs for DOPs has attracted a growing interest due to its importance in EA's real world applications [7]. The simplest way of addressing DOPs is to restart EAs from scratch whenever an environment change is detected. Though the restart scheme really works for some cases [8], for many DOPs it is more efficient to develop other approaches that make use of knowledge gathered from old environments. One of the possible approaches is to maintain and reintroduce diversity during the run of EAs, i.e., the immigrants schemes [9], [10], [11]. Elitism-based immigrants scheme [12] is a representative one among immigrants schemes for EAs in dynamic environments. In the scheme, the elite from previous generation is used as the base to create immigrants via mutation to replace the worst individuals in the current population. This way, the introduced immigrants are more adapted to the changing environment.

In this paper, we implement and apply the elitism-based immigrants GA to solve the dynamic SP problem. First, we design the specific genetic algorithm for the dynamic SP problem. Then at each generation, a certain number of elitism-based immigrants are generated and added into the population to maintain the diversity. Once the topology is changed, the new immigrants can help guide the search of good solutions in the new environment. Since end-to-end delay [13] is a pretty important quality-of-service (QoS) metric to guarantee the real-time data delivery, we also require the routing path to satisfy the delay constraint. For comparison purposes, we also implement the standard GA (SGA) and the Restart GA. By simulation experiments, we evaluate their performance on the dynamic SP problem. The results show that the EIGA significantly outperforms other two GA methods. It is verified that the EIGA works really well in the dynamic real-world networks.

H. Cheng and S. Yang are with the Department of Computer Science, University of Leicester, University Road, Leicester LE1 7RH, UK (phone: 44-116-2525295; fax: 44-116-2523915; email: hc118@le.ac.uk).

II. Related Work

The SP problem has been investigated extensively. Since the algorithms with polynomial time complexity are not suitable for the real-time computation of shortest paths, quite a few research work have been conducted to solve the SP problems using artificial intelligence techniques, e.g., ANNs [5], GAs [6], and PSO [14].

In [5], a near-optimal routing algorithm employing a modified Hopfield neural network (HNN) is proposed. It uses every piece of information that is available at the peripheral neurons, in addition to the highly correlated information that is available at the local neuron. Therefore, it can achieve faster convergence and better route optimality than other HNN based algorithms. In [6], a genetic algorithmic approach is presented to the SP routing problem. Computer simulations show that the GA based SP algorithm exhibits a much better quality of solution (route optimality) and a much higher rate of convergence than other algorithms. It also develops a population-sizing equation that facilitates a solution with desired quality. In [14], a PSO-based search algorithm is proposed. A priority-based indirect path-encoding scheme is used to widen the scope of search space and a heuristic operator is used to reduce the probability of invalid loop creation during the path construction procedure. It claims that the PSO-based SP algorithm is superior to those using GAs including the one in [6].

However, all these algorithms still address the static SP problem only. When the network topology changes, they will regard it as a new network and restart the algorithms over the new topology. As is well known that the topology changes rapidly in MANETs due to the characteristics of wireless networks, e.g., battery exhaustion, node mobility. Therefore, for the dynamic SP problem in MANETs, these algorithms are not good choices since they require frequent restart and cannot meet the real-time requirement. In this regard, EIGA has its inherent advantage, that is, it uses the immigrants to help the population quickly adapt to the new environment after the change occurs. Hence, the algorithm can keep running over the continuously changing topologies and avoid the expensive and inefficient restart. Regarding EIGA, to our best knowledge, we are not aware of any applications to the real-world problems.

III. Model

In this section, we first present our network model and then formulate the problem of dynamic SP routing.

We consider a mobile ad hoc network operating within a fixed geographical region. We model it by a undirected and connected topology graph $G_0(V_0, E_0)$, where V_0 represents the set of wireless nodes (i.e., routers) and E_0 represents the set of communication links connecting two neighboring routers falling into the radio transmission range. A communication link (i, j) can not be used for packet transmission until both node i and node j have a radio interface each with a common channel. However, the channel assignment is beyond the scope of this paper. In addition, message transmission on a wireless communication link will incur remarkable delay and cost.

Here, we summarize some notations that we use throughout this paper.

- $G_0(V_0, E_0)$, the initial MANET topology graph.
- $G_i(V_i, E_i)$, the MANET topology graph after the ith change.
- s, the source node.
- r, the destination node.
- $P_i(s, r)$, a path from s to r on the graph G_i.
- d_l, the transmission delay on the communication link l.
- c_l, the cost on the communication link l.
- $\Delta(P_i)$, the total transmission delay on the path P_i.
- $C(P_i)$, the total cost of the path P_i.

The problem of the dynamic SP routing can be informally described as follows. Initially, given a network of wireless routers, a delay upper bound, a source node and a destination node, we wish to find a delay-bounded least cost loop-free path on the topology graph. Then periodically or stochastically, due to energy conservation or some other issues, some nodes are scheduled to sleep or some sleeping nodes are scheduled to wake up. Therefore, the network topology changes from time to time. The objective of our problem is to quickly find the new optimal delay-constrained least cost acyclic path after each topology change.

More formally, consider a mobile ad hoc network $G(V, E)$ and a unicast communication request from the source node s to the destination node r with the delay upper bound Δ. The *dynamic delay-constrained shortest path problem* is to find a series of paths $\{P_i | i \in \{0, 1, ...\}\}$ over a series of graphs $\{G_i | i \in \{0, 1, ...\}\}$, which satisfy the delay constraint as shown in (1) and have the least path cost as shown in (2).

$$\Delta(P_i) = \sum_{l \in P_i(s,r)} d_l \le \Delta \,. \tag{1}$$

$$C(P_i) = \min_{P \in G_i} \left\{ \sum_{l \in P(s,r)} c_l \right\} . \tag{2}$$

IV. Design of GA for SP problem

This section describes the design of the GA for the SP problem. The GA operations consist of several key components: genetic representation, population initialization, fitness function, selection scheme, crossover and mutation. A routing path consists of a sequence of adjacent nodes in the network. Hence, it is a natural choice to adopt the path-oriented encoding method. For the routing problems, the path-oriented encoding and the path-based crossover and mutation are also very popular [6], [15].

A. Genetic Representation

A routing path is encoded by a string of positive integers that represent the IDs of nodes through which the path passes. Each locus of the string represents an order of a node (indicated by the gene of the locus). The gene of the first locus is for the source node and the one of the last locus is for the destination node. The length of a routing path should not exceed the maximum length $|V_0|$, where V_0 is the set of nodes in the MANET. Chromosomes are encoded under the delay constraint. In case it is violated, the encoding process is usually repeated so as to satisfy the delay constraint.

B. Population Initialization

In GA, each chromosome corresponds to a potential solution. The initial population Q is composed of a certain number, denoted as q, of chromosomes. To explore the genetic diversity, in our algorithm, for each chromosome, the corresponding routing path is randomly generated. We start to search a random path from s to r by randomly selecting a node v_1 from $N(s)$, the neighborhood of s. Then we randomly select a node v_2 from $N(v_1)$. This process is repeated until r is reached. Thus, we get a random path $P(s, r)=\{s, v_1, v_2, ..., r\}$. Since the path should be loop-free, the nodes that are already included in the current path are excluded, thereby avoiding reentry of the same node. The initial population is generated as follows.

Step 1: Start(i=0).
Step 2: Generate chromosome Ch_i: search a random loop-free path $P(s, r)$;
Step 3: i=i+1. If $i < q$, go to *Step 2*, otherwise, stop.
Thus, the initial population $Q = \{Ch_0, Ch_1, ..., Ch_{q-1}\}$ is obtained.

C. Fitness Function

Given a solution, we should accurately evaluate its quality (i.e., fitness value), which is determined by the fitness function. In our algorithm, we aim to find the least cost path between the source and the destination. Our primary criterion of solution quality is the path cost. Therefore, among a set of candidate solutions (i.e., unicast paths), we choose the one with the least path cost. The fitness value of chromosome Ch_i (representing the path P), denoted as $F(Ch_i)$, is given by:

$$F(Ch_i) = [\sum_{l \in P(s,r)} c_l]^{-1} . \tag{3}$$

The proposed fitness function only involves the total path cost. As mentioned above, The delay constraint is checked for each chromosome in the course of the run.

D. Selection Scheme

Selection plays an important role in improving the average quality of the population by passing the high quality chromosomes to the next generation. The selection of chromosome is based on the fitness value. We adopt the scheme of pair-wise tournament selection without replacement [16] as it is simple and effective. The tournament size is 2.

E. Crossover and Mutation

Genetic algorithm relies on two basic genetic operators - crossover and mutation. Crossover processes the current solutions so as to find better ones. Mutation helps GA keep away from local optima [6]. Performance of GA very depends on them. Type and implementation of operators depends on encoding and also on a problem.

In our algorithm, since chromosomes are expressed by the path structure, we adopt single point crossover to exchange partial chromosomes (subpath) at positionally independent crossing sites between two chromosomes [6]. With the crossover probability, each time we select two chromosomes Ch_i and Ch_j for crossover. Ch_i and Ch_j should possess at least one common node. Among all the common nodes, one node, denoted as v, is randomly selected. In Ch_i, there is a path consisting of two parts: ($s \xrightarrow{Ch_i} v$) and ($v \xrightarrow{Ch_i} r$). In Ch_j, there is a path consisting of two parts: ($s \xrightarrow{Ch_j} v$) and ($v \xrightarrow{Ch_j} r$). The crossover operation exchanges the subpaths ($v \xrightarrow{Ch_i} r$) and ($v \xrightarrow{Ch_j} r$).

The population will undergo the mutation operation after the crossover operation is performed. With the mutation probability, each time we select one chromosome Ch_i on which one gene is randomly selected as the mutation point (i.e., mutation node), denoted as v. The mutation will replace the subpath ($v \xrightarrow{Ch_i} r$) by a new random subpath.

Both crossover and mutation may produce new chromosomes which are infeasible solutions. Therefore, we check if the paths represented by the new chromosomes are acyclic. If not, repair functions [17] will be applied to eliminate the loops. Here the detail is omitted due to the space limit. All the new chromosomes produced by crossover or mutation satisfy the delay constraint since it has already been taken into consideration.

V. Elitism-based Immigrants GA

In stationary environments, convergence at a proper pace is really what we expect for GAs to locate the optimum solutions for many optimization problems. However, for DOPs, convergence usually becomes a big problem for GAs because changing environments usually require GAs to keep a certain population diversity level to maintain their adaptability. To address this problem, the random immigrants approach is a quite natural and simple way [18], [19]. It was proposed by Grefenstette with the inspiration from the flux of immigrants that wander in and out of a population between two generations in nature. It maintains the diversity level of the population through replacing some individuals of the current population with random individuals, called random immigrants, every generation. As to which individuals in the population should be replaced, usually there are two strategies: replacing random individuals or replacing the worst ones [20]. In order to avoid that random immigrants disrupt the ongoing search progress too much, especially during the period when the environment does not change, the ratio of the number of random immigrants to the population size is usually set to a small value, e.g., 0.2.

However, in a slowly changing environment, the introduced random immigrants may divert the searching force of the GA during each environment before a change occurs and hence may degrade the performance. On the other hand, if the environment only changes slightly in terms of severity of changes, random immigrants may not have any actual effect even when a change occurs because individuals in the previous environment may still be quite fit in the new environment. Based on the above consideration, an immigrants approach, called elitism-based immigrants [12], is proposed for GAs to address DOPs.

The pseudo-code for the EIGA is shown below. Within EIGA, for each generation t, after the normal genetic operations (i.e., selection and recombination), the elite $E(t\text{-}1)$ from previous generation is used as the base to create immigrants. From $E(t\text{-}1)$, a set of $r_{ei}\times n$ individuals are iteratively generated by mutating $E(t\text{-}1)$ with a probability p_m^i, where n is the population size and r_{ei} is the ratio of the number of elitism-based immigrants to the population size. The generated individuals then act as immigrants and replace the worst individuals in the current population. It can be seen that the elitism-based immigrants scheme combines the idea of elitism with traditional random immigrants scheme. It uses the elite from previous population to guide the immigrants toward the current environment, which is expected to improve GA's performance in dynamic environments.

```
begin
  t:=0 and initialize population P(0) randomly
  evaluate population P(0)
  repeat
    P´(t)=selectForReproduction(P(t))
    crossover(P´(t), $p_c$) // $p_c$ is the crossover probability
    mutate(P´(t), $p_m$) // $p_m$ is the mutation probability
    evaluate the interim population P´(t)

    // perform elitism-based immigration
    denote the elite in P(t-1) by E(t-1)
    generate $r_{ei}\times n$ immigrants by mutating E(t-1) with $p_m^i$
    evaluate these elitism-based immigrants

    replace the worst individuals in P´(t) with the generated
    immigrants
    P(t+1):=P´(t)
  until the termination condition is met // e.g., $t > t_{max}$
end
```

In our implementation of EIGA, if the mutation probability p_m^i is satisfied, the elite $E(t\text{-}1)$ will be used to generate the new immigrants by the mutation operation; otherwise, $E(t\text{-}1)$ itself will be directly used as the new immigrants.

VI. Experimental Study

We implement the EIGA, the SGA, and the Restart GA for the dynamic SP problem. For EIGA and SGA, if the change makes one individual in the current population become infeasible (e.g., one link in the corresponding path is lost after the change), we add penalty value to that individual. By simulation experiments, we evaluate their performance in a continuously changing mobile ad hoc network.

A. Experimental Design

The initial network topology is generated using the following method. We first specify a square region with the area of 200×200 that has the width [0, 200] on the x axis and the height [0, 200] on the y axis. Then we generate 100 nodes and the position (x, y) of each node is randomly specified within the square area. If the distance between two nodes falls into the radio transmission range D, a link will be added to connect them and both the cost and the delay of this link are randomly assigned within the corresponding ranges. Finally, we check if the generated topology is connected. If not, the above process is repeated until a connected topology is generated. In the experiments, D is given a reasonable value 50.

All the algorithms start from the initial network topology. Then after a certain number (saying, R) of generations (i.e., the change interval), a certain number (saying, M) of nodes are scheduled to sleep or wake up depending on their current status. It means that the selected working nodes will be turned off to sleep and the selected sleeping nodes will be turned on to work. Therefore, the network topology is changed accordingly since some links are lost and some other links appear again. By this means, we create a series of network topologies corresponding to the continuous network changes. Furthermore, these adjacent topologies are highly related since each time the changes affect only part of the nodes. We can see that R and M determine the change frequency and severity, respectively. The larger the value of R, the slower the changes. The larger the value of M, the more severe the changes. Currently, we set M to be 2.

As described in Section IV.D, the GA adopts pair-wise tournament selection without replacement. In all the experiments, the mutation probability is set to 0.1. For the elitism-based immigrants scheme, r_{ei} is set to 0.2 and p_m^i is set to 0.8. In addition, we set the number of changes to 10 and therefore the algorithms will work over 11 different but highly-related network topologies (the initial topology plus the 10 changed topologies). Both the source and destination nodes are randomly selected and they are not allowed to be scheduled in any change. The delay upper bound Δ is set to be 2 times of the minimum end-to-end delay.

B. Experimental Results and Analysis

At each generation, for each algorithm, we select the best individual from the current population and output the cost of

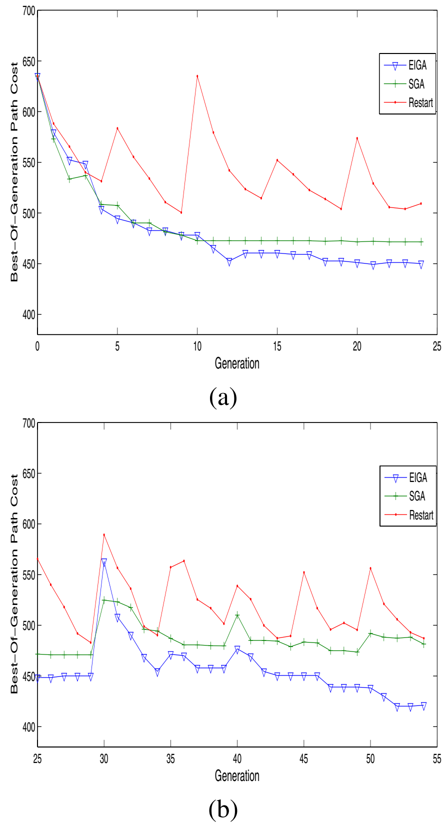


Fig. 1. Comparison of EIGA, SGA, and Restart GA when Z is 20 and R is 5: (a) generation 0 to 24; (b) generation 25 to 54.

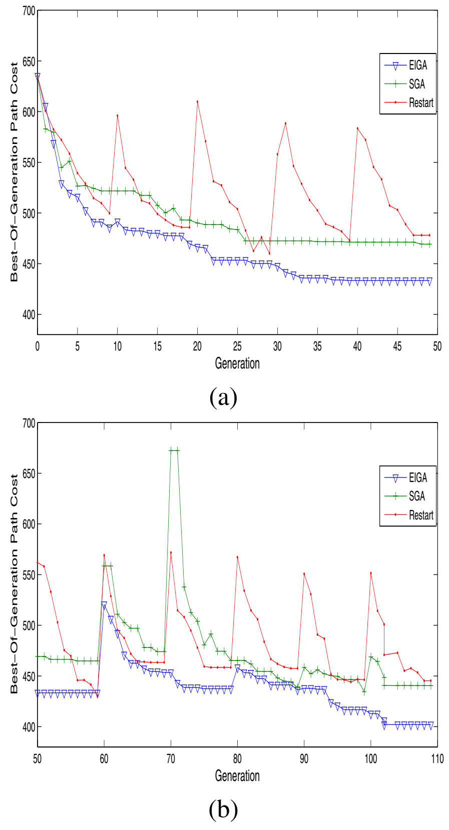


Fig. 2. Comparison of EIGA, SGA, and Restart GA when Z is 20 and R is 10: (a) generation 0 to 49; (b) generation 50 to 109.

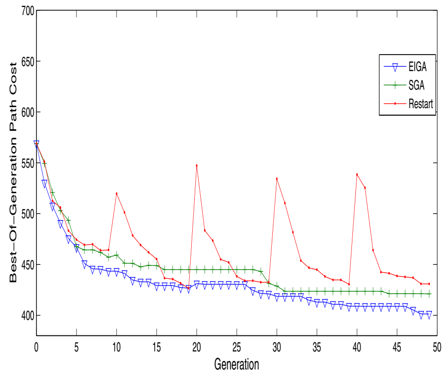
the shortest path represented by it. We repeat each experiment 10 times and get the average values of the best solutions at each generation. We set R to 5 and 10, respectively to see the impact of change frequence on the performance. When R is 10, we also vary the population size Z from 20 to 50 to see if an adequate population size can be determined in this problem.

Fig. 1 is the comparison results when the population size Z is 20 and the change interval R is 5. In Fig. 2, we change R to 10. Therefore, Fig. 1 shows a rapidly changing environment and Fig. 2 shows a relatively slowly changing environment. In Fig. 3, the change interval R is 10 while the population size Z is increased to 50. In all the three settings, we can see that both EIGA and SGA experience more significant changes in subfig (b) than in subfig (a). The reason is that when more nodes are rescheduled, the changes to the initial network topology become more drastic. In subfig (a), it seems that the changes do not involve the network nodes encoded in the best individual of the current population. Therefore, both EIGA and SGA keeps evolving while EIGA can find better solutions than SGA. The elitism-based immigrants bring more diversity to the population in EIGA and therefore enhance its search capability. However, from all the figures, we can see that the Restart GA exhibits the worst performance even when the changes have trivial impacts on the current population. The reason is that the Restart GA does not exploit any useful information in the old environment and that the frequent restart sacrifices its evolving capability.

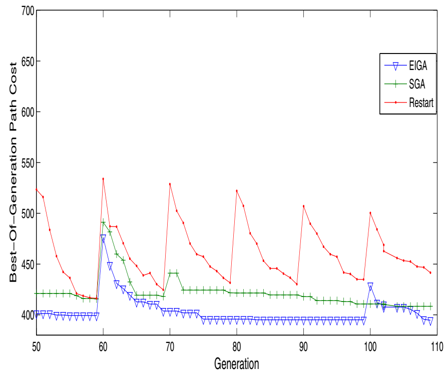
As shown in the three subfig (b), the changes start to affect the best individual in the old environment. In a relatively slowly changing environment, EIGA shows better performance than in a rapidly changing environment since it has more time to search good solutions before a change occurs. So does SGA. When a change occurs, Fig. 2 shows that EIGA can quickly adapt to the new topology and find good solutions again before next change occurs. Although SGA also shows the ability to adapt the population to the new environment, the best solutions that it can find in the new environment are not competitive. By comparing Fig. 2 with Fig. 3, we can see that when the population size is increased to 50, all the three algorithms show better performance than before. It is because that larger populations produce higher chances to find better solutions. Furthermore, EIGA adapts to the new environment more quickly since more immigrants are introduced. In summary, EIGA shows the best performance for the dynamic SP problem and SGA is better than Restart GA.

VII. Conclusions

The static SP problem considers the fixed network topology only. Intuitively, it is much more challenging to deal



(a)



(b)

Fig. 3. Comparison of EIGA, SGA, and Restart GA when Z is 50 and R is 10: (a) generation 0 to 49; (b) generation 50 to 109.

with the dynamic SP problem in a continuously changing network such like MANETs than to solve the static one in a fixed infrastructure. In recent years, there has been a growing interest in studying GAs for dynamic optimization problems. Among approaches developed for GAs to deal with DOPs, immigrants schemes for GAs on DOPs aim at maintaining the diversity of the population throughout the run via introducing new individuals into the current population. In this paper, we propose the EIGA to solve the dynamic SP problem in a large scale MANET. We well design the GA components for the SP problem and the elitism-based immigrants scheme. Simulation experiments show that EIGA is a powerful technique for solving the dynamic SP problem. We believe that this is the first work to well investigate the effectiveness and efficiency of the dynamic EA schemes in solving the dynamic problems in the real-world networks.

Acknowledgment

This work was supported by the Engineering and Physical Sciences Research Council (EPSRC) of UK under Grant EP/E060722/1.

References

[1] C. E. Perkins, Editors, *Ad Hoc Networking*, London: Addison-Wesley, 2001.

[2] C.-K. Toh, *Ad Hoc Mobile Wireless Networks: Protocols and Systems*, Prentice Hall PTR, 2002.

[3] C. Siva Ram Murthy and B. S. Manoj, *Ad Hoc Wireless Networks: Architectures and Protocols*, Prentice Hall PTR, 2004.

[4] M. K. Ali and F. Kamoun, "Neural networks for shortest path computation and routing in computer networks," *IEEE Transactions on Neural Networks*, vol. 4, no. 6, pp. 941–954, Nov. 1993.

[5] C. W. Ahn, R. S. Ramakrishna, C. G. Kang, and I. C. Choi, "Shortest path routing algorithm using hopfield neural network," *Electronics Letters*, vol. 37, no. 19, pp. 1176–1178, Sept. 2001.

[6] C. W. Ahn and R. S. Ramakrishna, "A genetic algorithm for shortest path routing problem and the sizing of populations," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 6, pp. 566–579, Dec. 2002.

[7] S. Yang and X. Yao, "Population-based incremental learning with associative memory for dynamic environments," *IEEE Transactions on Evolutionary Computation*, vol. 12, no. 5, pp. 542–561, Oct. 2008.

[8] S. Yang and X. Yao, "Experimental study on population-based incremental learning algorithms for dynamic optimization problems," *Soft Computing*, vol. 9, no. 11, pp. 815–834, Nov. 2005.

[9] S. Yang and R. Tinos, "A hybrid immigrants scheme for genetic algorithms in dynamic environments," *International Journal of Automation and Computing*, vol. 4, no. 3, pp. 243–254, July 2007.

[10] R. Tinos and S. Yang, "A self-organizing random immigrants genetic algorithm for dynamic optimization problems," *Genetic Programming and Evolvable Machines*, vol. 8, no. 3, pp. 255–286, Sept. 2007.

[11] S. Yang, "Genetic algorithms with memory- and elitism-based immigrants in dynamic environments," *Evolutionary Computation*, vol. 16, no. 3, pp. 385–416, Sept. 2008.

[12] S. Yang, "Genetic algorithms with elitism-based immigrants for changing optimization problems," *Proc. Fourth European Workshop on Evolutionary Algorithms in Stochastic and Dynamic Environments*, Valencia, Spain, Apr. 2007, pp. 627–636.

[13] M. Parsa, Q. Zhu, and J. Garcia-Luna-Aceves, "An iterative algorithm for delay-constrained minimum-cost multicasting," *IEEE/ACM Transactions on Networking*, vol. 6, no, 4, pp. 461–474, Aug. 1998.

[14] A. W. Mohemmed, N. C. Sahoo, and T. K. Geok, "Solving shortest path problem using particle swarm optimization," *Applied Soft Computing*, vol. 8, no. 4, pp. 1643–1653, Sept. 2008.

[15] D. Din, "Anycast routing and wavelength assignment problem on WDM network," *IEICE Transactions on Communications*, vol. E88-B, no. 10, pp. 3941–3951, Oct. 2005.

[16] S. Lee, S. Soak, K. Kim, H. Park, and M. Jeon, "Statistical properties analysis of real world tournament selection in genetic algorithms," *Applied Intelligence*, vol. 28, no. 2, pp. 195–205, Apr. 2008.

[17] S. Oh, C. Ahn, and R. Ramakrishna, "A genetic-inspired multicast routing optimization algorithm with bandwidth and end-to-end delay constraints," *Proc. 13th International Conference on Neural Information Processing*, Hong Kong, China, Oct. 2006, pp. 807–816.

[18] J. J. Grefenstette, "Genetic algorithms for changing environments," *Proc. 2nd International Conference on Parallel Problem Solving from Nature*, Brussels, Belgium, Sep. 1992, pp. 137–144.

[19] H. G. Cobb and J. J. Grefenstette, "Genetic algorithms for tracking changing environments," *Proc. 5th International Conference on Genetic Algorithms*, Urbana-Champaign, IL, Jun. 1993, pp. 523–530.

[20] F. Vavak and T. C. Fogarty, "A comparative study of steady state and generational genetic algorithms for use in nonstationary environments," *Proc. AISB Workshop on Evolutionary Computing*, Brighton, UK, Apr. 1996, pp. 297–304.