GA-based heuristic algorithms for bandwidth-delay-constrained least-cost multicast routing

A. T. Haghighat[***], K. Faez[*], M. Dehghan[**], A. Mowlaei[*], Y. Ghahremani

[*]Dept. of Electrical Engineering, Amirkabir University of Technology, Tehran 15914, Iran
[**]Iran Telecommunication research Center (ITRC), Tehran, Iran[♣]
[***]Atomic Energy Organization of Iran (AEOI) , Tehran, Iran

Tel: +98-21-6466009 Fax: +98-21-6406469
Email :kfaez@atu.ac.ir

Abstract

The bandwidth-delay-constrained least-cost multicast routing is a challenging problem in high-speed multimedia networks. Computing such a constrained Steiner tree is an NP-complete problem. In this paper, we propose a novel QoS-based multicast routing algorithm based on the genetic algorithms (GA). In the proposed method, the connectivity matrix of edges is used for genotype representation. Some novel heuristic algorithms are also proposed for mutation, crossover, and creation of random individuals. We evaluate the performance and efficiency of the proposed GA-based algorithm in comparison with other existing heuristic and GA-based algorithms by the result of simulation. This proposed algorithm has overcome all of the previous algorithms in the literatures.

Keywords: *Quality of Service, Multicast routing, Multi-constrained Steiner tree, Genetic algorithm*

1. Introduction

Recently, advances in media technologies such as optical fiber and switch technologies such as ATM and MPLS have resulted in a new generation of gigabit-per-second wide area networks. These networks are expected to support a wide range of communication-intensive real-time multimedia applications like digital audio and video. The deployment of high-speed networks opens a new dimension of research, providing quality of service (QoS) such as guaranteed throughput for video-on-demand application, low end-to-end delay for video conferencing, less than 200 ms end-to-end delay and low cell loss ratio for real-time audio applications, and high transmission reliability for distributed control applications. It is technically a challenging and complicated problem to deliver multimedia information in a timely, smooth, synchronized manner over a decentralized, shared network environment, especially one that was originally designed for best-effort traffic such as Internet.

In the past, most of the applications were unicast in nature and none of them had any QoS requirements. Therefore, the routing algorithms were very simple. However, with emerging distributed real-time multimedia applications such as video conferencing, distance learning, and video on demand, the situation is completely different now. These applications will involve multiple users, with their own different QoS requirements in terms of throughput, reliability, and bounds on end-to-end delay, jitter, and packet loss ratio. Accordingly, a key issue in the design of broad-band architectures is how to efficiently manage the resources in order to meet the QoS requirements of each connection. The establishment of efficient QoS routing schemes is, undoubtedly, one of the major building blocks in such architectures. Supporting point to multi-point connections for multimedia applications requires the development of efficient multicast routing algorithms. Multicast employs a tree structure of the network to efficiently deliver the same data stream to a group of receivers. In multicast routing, one or more constraints must be applied to the entire tree. Several well-known multicast routing problems have been studied in the literatures. The Steiner tree problem [1] tries to find the least-cost tree, the tree covering a group of destinations with the minimum total cost over all the links. It is also called the least-cost multicast routing problem, belonging to the class of tree-optimization problems. Finding either a Steiner tree or a constrained Steiner tree is NP (Non-deterministic Polynomial)-complete [2]. In this paper, we consider a bandwidth-delay-constrained least-cost multicast routing. For the purpose of clarity, in this paper we assume an environment where a source node is presented with a request to establish a new least-cost tree with two constrained: bandwidth constraint in all the links of the tree and end-to-end delay constraint from the source node to each of the

♣ This work was supported by ITRC under grant number 81-7234

destinations. In other words, we consider the source routing strategy, in which each node maintains the complete global state of the network, including the network topology and state information of each link. Most of the proposed algorithms for Steiner tree (without constraint) are heuristic. Some of the well-known Steiner tree heuristics are the RS heuristic [8], the TM heuristic [9], and the KMB heuristic [7]. Several algorithms based on neural networks [10] and genetic algorithms (GA) [23-27] have been also proposed for solving this problem.

Recently, a lot of delay-constrained least-cost multicast routing heuristics such as the KPP heuristic [4], the BSMA heuristic [3] and so on ([5], [6], and [11]) have been proposed. However, the simulation results given by Salama et al. [17] have shown that most of the heuristic algorithms either work too slowly or can not compute delay-constrained multicast tree with least cost. The best deterministic delay constraint low-cost (near optimal) algorithm is BSMA ([17], [28], [32]). Note that the above algorithms have designed specifically for real-time applications with only one QoS constraint without mentioning how to extend these algorithms to real-time applications with two or more QoS constraints.

Since deterministic heuristic algorithms for QoS multicast routing are usually very slow, methods based on computational intelligence such as neural networks and genetic algorithms may be more suitable.

Chotipat et al. [18] have been proposed an algorithm based on Hopfield neural network to solve QoS multicast routing. However, the selection of the coefficients in energy (or Lyapunov) function is complex and sometimes may lead to unexpected wrong solution. In addition, because of using a continuous Hopfield neural network, the QoS routing solutions must be assumed to be continuous, which makes the problem more complex.

In the field of computational intelligence, GA-based algorithms have emerged as powerful tools for solving NP-complete constrained optimization problems. Several GA-based algorithms [23-27] have been proposed for solving Steiner tree problem without QoS constraints. Also, Sun [28] has extended the algorithm proposed in [26] for the least-cost multicast routing problem with one QoS constraint (delay). For deploying the genotype encoding used in [26], [28], another NP-complete sub-problem (a deterministic delay-constrained least-cost multicast routing algorithm, CKMB [11]) must be solved during the decoding phase. Furthermore, the algorithm assumes the same delay constraints for all destinations, which greatly restricts its application. However, the simulation results given by Sun have shown that his algorithm can achieve trees with smaller average cost than those of BSMA, in a shorter running time for relatively large networks. Xiang et al. [29] have proposed a GA-based algorithm for QoS routing in general case. This algorithm adopts an $N * N$ one-dimensional binary encoding scheme, where N represents the number of nodes in the graph. However, in this encoding scheme, the transformation back and forth between genotype and phenotype space is very complicated, especially for large networks. Ravikumar et al. [30] have proposed a GA-based algorithm with novel interesting approaches for crossover and mutation operators for the delay-constrained least-cost multicast routing problem. However, they have not defined their scheme for encoding and decoding of individuals. Since their algorithm may lead to premature convergence, an approach must be designed to prevent this phenomenon [33]. Zhang et al. [31] have proposed an effective orthogonal GA for delay-constrained least-cost multicast routing problem. This algorithm also assumes the delay constraints for all destinations are identical. Also, Wu et al. [32] have proposed a GA-based algorithm for multiple QoS constraints multicast routing problem in general case. However, their proposed genotype representation does not necessarily represent a tree. On the other hand, It is necessary to construct and store a very large amount of possible routes for each pairs of nodes in the graph using the K-shortest path algorithm. Wang et al. [33] have proposed an efficient GA-based heuristic algorithm for bandwidth-delay-constrained least-cost multicast routing problem. They have used a tree data structure for genotype representation, but not clearly defined their encoding and decoding schemes.

In this paper, we propose a novel QoS-based multicast routing algorithm based on genetic algorithms (GA). In the proposed method, the connectivity matrix of edges is used for genotype representation. Some novel heuristic algorithms are also proposed for mutation, crossover, and creation of random individuals. We evaluate the performance and efficiency of the proposed GA-based algorithm in comparison with other existing heuristic and GA-based algorithms by the result of simulation. This proposed algorithm has overcome all of the previous algorithms in the literatures.

The remainder of this paper is organized as follows. The problem description and formulation is given in section 2. In Section 3, we describe the proposed algorithms. We then evaluate the convergence of the proposed GA-based algorithms in Section 4. Section 5, gives the performance evaluation of the proposed algorithms and the comparison of them with other similar algorithms. Section 6 concludes this study and discusses future works.

2. Problem description and formulation

A network is modeled as a directed, connected graph $G = (V, E)$, where V is a finite set of *vertices* (network nodes) and E is the set of *edges* (network links) representing connection of these vertices. Let $n = |V|$ be the number of network nodes and $l = |E|$ be the number of network links. The link $e = (u, v)$ from node $u \in V$ to node $v \in V$ implies the existence of a link $e' = (v, u)$ from node v to node u. Three non-negative real value functions are associated with each link e ($e \in E$): cost $C(e): E \to R^+$, delay $D(e): E \to R^+$, and available bandwidth $B(e): E \to R^+$. The link cost function, $C(e)$, may be either monetary cost or any measure of the resource utilization, which must be optimized. The link delay, $D(e)$, is considered to be the sum of switching, queuing, transmission , and propagation delays. The link bandwidth, $B(e)$, is the residual bandwidth of the physical or logical link. The link delay and bandwidth functions, $D(e)$ and $B(e)$, define the criteria that must be constrained (bounded). Because of the asymmetric nature of the communication networks, it is often the case that $C(e) \neq C(e')$, $D(e) \neq D(e')$, and $B(e) \neq B(e')$.

A multicast tree $T(s, M)$ is a sub-graph of G spanning the source node $s \in V$ and the set of destination nodes $M \subseteq V-\{s\}$. Let $m = |M|$ be the number of multicast destination nodes. We refer to M as the *destination group* and $\{s\} \cup M$ the *multicast group*. In addition, $T(s, M)$ may contain relay nodes (Steiner nodes), that is, the nodes in the multicast tree but not in the multicast group. Let $P_T(s, d)$ be a unique path in the tree T from the source node s to a destination node $d \in M$.

The total cost of the tree $T(s, M)$ is defined as the sum of the cost of all links in that tree and can be given by

$$C(T(s,M)) = \sum_{e \in T(s,M)} C(e)$$

The total delay of the path $P_T(s, d)$ is simply the sum of the delay of all links along $P_T(s, d)$:

$$D(P_T(s,d)) = \sum_{e \in P_T(s,d)} D(e)$$

The bottleneck bandwidth of the path $P_T(s, d)$ is defined as the minimum available residual bandwidth at any link along the path:

$$B(P_T(s,d)) = \min\{B(e), e \in P_T(s,d)\}$$

Let Δ_d be the delay constraint and B_d the bandwidth constraint of the destination node d. The bandwidth-delay-constrained least-cost multicast problem is defined as minimization of $C(T(s, M))$ subject to

$$\begin{cases} D(P_T(s,d)) \leq \Delta_d, \forall d \in M \\ B(P_T(s,d)) \geq B_d, \forall d \in M \end{cases}$$

3. The proposed GA-based algorithms

Genetic algorithms, as powerful and broadly applicable stochastic search and optimization techniques, are the most widely known types of evolutionary computation methods today. In general, a genetic algorithm has five basic components as follows: 1) An encoding method, that is a genetic representation (genotype) of solutions to the program. 2) A way to create an initial population of individuals (chromosomes). 3) An evaluation function, rating solutions in terms of their fitness and a selection mechanism. 4) The genetic operators (crossover and mutation) that alter the genetic composition of offspring during reproduction. 5) Values for the parameters of genetic algorithm. A general structure of the genetic algorithms is as follows:

```
Procedure: Genetic Algorithms
Begin
  t := 0;
  initialize P(t);      {P(t) is the population of individuals in generation t}
  evaluate P(t);
  While (not termination condition) do
  Begin
     recombine P(t) to yield C(t);     {creation of offspring C(t) by means of genetic operators}
     evaluate C(t);
     select P(t + 1) from P(t) and C(t);
     t := t + 1;
  End
End
```

Figure 1: General structure of the genetic algorithms

3.1. Genotype

Let us define the connectivity matrix of edges, Y_{n*n}, such that the value of each element ($Y[i, j] \in \{0, 1\}$) tells whether or not a specific edge connects the pair of nodes (i, j). For converting the connectivity matrix Y into a one-dimensional chromosome x, which consists of $n*(n-1)/2$ elements, we should transfer the elements on the top triangle of matrix Y, from the first row and from left to right into the chromosome x as indicated in the Figure 2. Although, we consider that the network is asymmetric, It is not necessary to use all elements of the connectivity matrix of edges to represent the Steiner tree. In other word, the top triangle of the connectivity matrix of edges is sufficient to represent the Steiner tree. Note that if $x[k] = Y[i, j]$, then the index k is represented as a function of i , j by the following equation:

$$k = \frac{n(n-1)}{2} - \frac{(n-i)(n-i+1)}{2} + (j-i)$$

```
Procedure: Connectivity matrix decoding
Begin
  For I := 1 to n do
    For j := i + 1 to n do
        Temp[i, j] := Y[i, j];
  Current-vertex := s;          {Select the source node s (root vertex) as the current vertex}
  k := 1;
  Add s to the k-th (first) path-list;
  While (there is any item equal to one in Temp matrix) do
    Begin
        {Check for a successor vertex to the Current-vertex from left to right}
         v := 0;
        For i := 1 to Current-vertex - 1 do
            If (Temp[i, Current-vertex] = 1)  then
                Begin
                   v := i;
                  Temp[i, Current-vertex] := 0;
                   exit;
                End
        If (v = 0) then
             For i := Current-vertex + 1 to n do
                  If (Temp[Current-vertex, i] = 1)  then
                    Begin
                      v := i;
                     Temp[Current-vertex, i] := 0;
                      exit;
                    End
        If (v ≠ 0) then
            Begin
               Add v to the k-th path-list;
               Degree := 0;
                For i := 1 to v - 1 do
                      Degree := Degree + Y[i, v];
                For i := v+1 to n do
                      Degree := Degree + Y[v, i];
                If (Degree = 1) then          {if v is a leaf }
                   Begin
                       Copy k-th path-list to (k+1)-th path-list;
                       k := k+1;
                       Remove the last item of k-th path-list;
                   End
                Else
                   Current-vertex := v;
            End
        Else  If (Current-vertex is not s) then
            Begin
                Remove the last item of the k-th link list;
               Current-vertex := Predecessor vertex;    {the last item of k-th link list}
            End
    End
    k := k -1;      {Remove the last link list}
End
```

Figure 2: Connectivity matrix decoding algorithm

3.2. Pre-Processing Phase

Before starting the genetic algorithm, we can remove all the links, which their bandwidth are less than the minimum of all required thresholds (Min $\{B_d \mid \forall d \in M\}$). If in the refined graph, the source node and all the destination nodes are not in a connected sub-graph, this topology does not meet the bandwidth constraint. In this case, the source should negotiate with the related application to relax the bandwidth bound. On the other hand, if the source node and all the destination nodes are in a connected sub-graph, we will use this sub-graph as the network topology in our GA-based algorithms.

3.3. Initial population

The creation of the initial population in this study is based on the randomized depth-first search algorithm [30],[33]. We propose a modified randomized depth-first search algorithms for this purpose:

Random individual creation algorithm: In this algorithm, a linked list is constructed from the source node *s* to one of the destination nodes. Then, the algorithm continues from one of the unvisited destinations and at each node the next unvisited node is randomly selected until one of the nodes in the previous sub-tree (the tree that is constructed in the previous step) is visited. The algorithm terminates when all destination nodes have been mounted to the tree. The procedure of creation the initial population has been shown in Figure 3. This procedure must be called *pop-size* times to create the total of initial population.

```
Procedure: random individual creation
Begin
  n := 1;
  First := True;
  While (n<=Number of Destinations) do
   Begin
      Initialize the n-th link list;
      If (First) then
          Current-node := Source
      Else
          Current-node := One of unvisited Destinations;
      G_TM := Temporary matrix of the network graph;
      Add the Current-node to the n-th link list;
      Link-list-comp := False;
      While (Not Link-list-comp) do
        Begin
          k := Number of connected nodes to the Current-node in G_TM;
          If (k=0) then
            Begin
              Remove the Current-node in the n-th link list;
              Remove the link between the Current-node and the previous node in G_old;
              Current-node := previous node in the n-th link list;
              G_TM := G_old
            End
          Else
            Begin
              i := a random natural number in interval [1,k];
              Add the i-th node to the n-th link list;
              G_old := G_TM;
              Remove all links to the Current-node in G_TM;
              Current-node := the i-th node;
              If (First) then
                 If (Current-node is one of the destinations) then
                    Begin
                      Link-list-comp := True;
                      Make an individual by n-th link list;
                      n := n+1;
                      First := False;
                      Mark the found destination as a visited destination
                    End
              Else
                 If (the Current-node is a node in one of the previous link lists(for example j-th link list)) then
                    { if the Current-node has a connection to the source node, this link has higher priority}
                    Begin
                      n-th link list := j-th link list from the source node to found position + Inverse (n_th link list);
                      Link-list-comp := True;
```

```
                    Add the n-th link list to the individual;
                    n := n+1;
                    Mark this destination as a visited destination
                End
            End {Else}
        End {inner while}
    End {outer while}
End {procedure}
```

Figure 3: A modified depth-first search algorithm to create a random individual

3.4. Fitness function

The fitness function in our study is an improved version of the scheme proposed in [33]. We define the fitness function for each individual, the tree $T(s, M)$, using the penalty technique, as follows:

$$F(T(s,M)) = \frac{\alpha}{\sum_{e \in T(s,M)} C(e)} \prod_{d \in M} \phi(D(P(s,d)) - \Delta_d) \prod_{d \in M} \phi(B(P(s,d)) - B_d)$$

$$\phi(z) = \begin{cases} 1 & z \le 0 \\ \gamma & z > 0 \end{cases}$$

Where α is a positive real coefficient, $\phi(z)$ is the penalty function and γ is the degree of penalty (γ is considered equal to 0.5 in our study). Wang et al. [33] have assumed that the bandwidth constraints (B_d) for all destinations are identical.

3.5. Selection

The selection process used here is based on spinning the roulette wheel *pop-size* times, and each time a single chromosome is selected as a new offspring. The probability P_i that a parent T_i is selected is given by:

$$p_i = \frac{F(T_i)}{\sum_{j=1}^{pop-size} F(T_j)}$$

Where $F(T_i)$ is the fitness of the T_i individual.

3.6. Crossover

Several crossover operators are described in the literatures [23-33] for Steiner tree and constrained Steiner tree problems. Some of them have used the traditional well-known crossover operators, such as the following schemes:

- One point crossover operator (e. g. see [28])
- One point crossover operator, with a fixed probability P_c(≈0.6-0.9) (e. g. see [27])
- Two point crossover operator (e. g. see [32])
- One point crossover operator plus "and" and "or" logic operations with a fixed probability P_c (see [29])

Unfortunately, according to the genotype representation in these papers, the above crossover operators are not suitable for recombination of two individuals (the crossover operation mostly leads to illegal individuals). However, Ravikumar et al. [30] have proposed a new interesting approach for crossover of Steiner trees and Wang et al. [33] have used the same scheme with some modifications. In this scheme, two multicast trees, $T_F(s, M)$ and $T_M(s, M)$, are selected as parents and the crossover operation produces an offspring $T_O(s, M)$ by identifying the links that are common to both parents. The operator selects the same links of two parents for quicker convergence of the genetic algorithm.

However, these common links may be in some separate sub-trees, and some edges may have to be added in order to transform them into a multicast tree. In this step, a multicast tree is constructed from these separate sub-trees. First, two separate sub-trees among these sub-trees are randomly selected, and are connected them with the least-delay or the least-cost path (in [30] all sub-trees are connected to the first sub-tree). If none of the parents satisfies the delay constraint, the least-delay path is chosen. Otherwise the least-cost path is chosen (in [30] this condition is checked for all individuals in the population). The path, which is added to join two sub-trees is selected heuristically. The two connected sub-trees is replaced with the new sub-tree in the sub-trees set. Next, conforming to the same rule, a new selection begins again. The selection

repeats until a multicast tree is constructed. Clearly there is no loop in the multicast tree constructed by this connection scheme. Finally, it may be possible that some of the leaf nodes of T_O are not the source node or destination nodes. These nodes are deleted from the offspring.

However, the first disadvantage of this scheme is the complexity of the heuristic algorithm, which selects a path to join the two separate sub-trees. The second disadvantage of this scheme is that the result of this complex heuristic algorithm is not necessarily a multicast tree including the source node and all destination nodes.

We propose two novel crossover schemes for recombination of two individuals, which represent Steiner trees:

Crossover I: Let $\{P_F(s, d_1), P_F(s, d_2), \ldots, P_F(s, d_m)\}$ be the set of paths from the source node s to all destination nodes in T_F and $\{P_M(s, d_1), P_M(s, d_2), \ldots, P_M(s, d_m)\}$ be the same set in T_M. Since, we have found these paths for all individuals in the current population for calculating the fitness function of them, the proposed algorithm will not be complex. We define a fitness function for the path $P(s, d_i)$ based on the total cost, the total delay and the minimum bandwidth of the path using the penalty technique, as follows:

$$F(P(s,d_i)) = \frac{\alpha}{\sum_{e \in P(s,d_i)} C(e)} \phi(D(P(s,d_i)) - \Delta_{d_i}) \phi(B(P(s,d_i)) - B_{d_i})$$

$$\phi(z) = \begin{cases} 1 & z \le 0 \\ \gamma & z > 0 \end{cases}$$

Where α is a positive real coefficient, $\phi(z)$ is the penalty function and γ is the degree of penalty (γ is considered equal to 0.5 in our study). According to the crossover probability of P_c, two multicast trees $T_F(s, M)$ and $T_M(s, M)$ are selected as parents and the crossover operation produce an offspring $T_O(s, M)$. Each individual may be recombined with its right individual and its left individual through the crossover operator. For each destination node d_i, we compute the fitness of $P_M(s, d_i)$ and $P_F(s, d_i)$ and select the better path. Finally, we compose all selected paths and construct a new Steiner tree (see Figure 4).

```
Procedure: The crossover operator
Begin
   For i:=1 to m do            { m is the number of destination nodes}
      If F(P_M(s, d_i)) > F(P_F(s, d_i)) then
         P_O(s, d_i) := P_M(s, d_i)
      Else
         P_O(s, d_i) := P_F(s, d_i);
   Current-tree := P_O(s, d_1);
   For i:=2 to m do
     Begin
        Previous-node := s;
        Start-node := s;
        Current-node := The second node in the P_O(s, d_i);
        New-link := False;
        While (Previous-node <> d_i) do
           Begin
              If the Current-node does not exist in the current-tree then
                 Begin
                    Add the link between the Current-node and the Previous node to the current-tree;
                    New-link := True;
                 End
              Else
                 Begin
                    If the New-link = True then
                       Remove all link from Start-node to the Previous-node in P_O(s, d_i) in the current-tree;
                    Start-node := Current-node
                    New-link := False;
                 End
              Previous-node := Current-node;
              If there is another node in P_O(s, d_i) then
                 Current-node := the next node in the P_O(s, d_i)
           End
     End
End
```

Figure 4: The proposed heuristic *crossover I* operator

Crossover II: In this scheme, we first use a simple one-point crossover operator, with a fixed probability P_c. The constructed offspring do not necessarily represent Steiner trees. Then, the effective and fast *check and recovery* algorithm proposed in [31] is used to connect the separate sub-trees in the offspring and also connecting the absent nodes of multicast group to the final tree.

3.7. Mutation

Many of proposed GA-based algorithms for multicast routing such as [27], [28], [29], and [32] have used the bit-flip mutation with a fixed small probability P_m(≈0.001-0.05). Unfortunately, according to the genotype representation in these papers, the bit mutation generates illegal individuals and decreases the performance of them. However, Ravikumar et al. [30] have proposed a new scheme for mutation of Steiner trees and Wang et al. [33] have used the improved version of it in their study. In this scheme [33], according to the mutation probability P_m, the mutation procedure randomly selects a subset of nodes and breaks the multicast tree into some separate sub-trees by removing all the links that are incident to the selected nodes. Then, it re-connects those separate sub-trees into a new multicast tree by randomly selecting the least-delay or the least-cost paths between them.

However, the result of this complex heuristic algorithm is not necessarily a multicast tree including the source node and all destination nodes. In this paper, we propose two following algorithms for mutation operator:

Mutation I: First, we propose an improved version of the scheme presented in [33]. The mutation procedure randomly selects a subset of nodes and breaks the multicast tree into some separate sub-trees by removing all the links that are incident to the selected nodes. Then, the effective and fast *check and recovery* algorithm proposed in [31] is used to connect the separate sub-trees and also connecting the absent nodes of multicast group to the final tree.

Mutation II: According to the mutation probability P_m, the mutation procedure randomly selects an infeasible chromosome from one of the following class (If the first class is empty, a chromosome is selected from the second class and so on)

- Class 1: The chromosomes, which do not satisfy the delay and the bandwidth constraints.
- Class 2: The chromosomes, which do not satisfy the delay constraint.
- Class 3: The chromosomes, which do not satisfy the bandwidth constraint.

If all chromosomes in the current population satisfy both of the QoS constraints, we exit from the mutation procedure. Then, we select only the paths that satisfy both of the QoS constraints in the selected chromosome. We re-connect these selected paths by our proposed algorithm of *crossover I* (see Figure 4). Finally, the disconnected destination nodes will be mounted to the sub-tree by our proposed algorithm of *random individual creation* (see Figure 3).

3.7. Illegality and Infeasibility

The chromosomes generated randomly in the initial population and the offspring produced by the mutation and crossover operators may be illegal or infeasible. Illegality refers to the phenomenon that a chromosome does not represent a multicast tree; Infeasibility refers to the phenomenon that a chromosome, which represents a multicast tree, does not satisfy the problem constraints. Three strategies have been proposed to deal with these violations:

- Rejecting strategy
- Penalizing strategy
- Repairing strategy

The penalty methods are mostly used to handle infeasible chromosomes [36]. We have used this strategy in our proposed fitness function. It is really difficult to provide a reasonable penalizing factor for the illegal chromosomes in our study, because the illegality can not be easily measured quantitatively. The repair strategy does indeed surpass other strategies, such as the rejecting or the penalizing strategies, in this case. We have used this strategy in our proposed *mutation I* and *crossover II* algorithms. On the other hand, we have proposed another strategy to deal with the illegality problem. We will refer to this strategy as the *avoidance strategy*. In this paper, most of the proposed algorithms, such as the initial population creation algorithm, the *crossover I* algorithm, and the *mutation II* algorithm, have been used this strategy to avoid creating illegal individuals.

4. Analysis of convergence

According the Theorem 2.7 in Ref. [34], the GA-based algorithms proposed in this paper could finally converge to the global optimal solution. For a large-scale network, it is time-consuming to obtain the optimal solution to the bandwidth-delay-constrained least-cost multicast routing problem, which is NP-complete. This problem can be overcome by setting an appropriate iteration time of the genetic algorithm. In this way, we can obtain a near-optimal solution within a reasonable time limit.

5. Experimental Results

In this section, we have used the simulation experiments to compare the performance of the proposed GA-based algorithms with the heuristic BSMA heuristic algorithm and some existing GA-based algorithms. We have implemented more than 2,000 lines C++ program to simulate all of the proposed algorithms. All simulation experiments are run on a Pentium III 800, 256 MB RAM, IBM PC. The experiments are run repeatedly until confidence interval of less than 5%, using 95% confidence level, are achieved for the simulation results. A random graph generator based on the Salama [17] graph generator is used. The average degree of each node in the random generated graphs is 4. The multicast group is randomly selected in the graph. The size of multicast group is considered 5%, 15%, and 25% of the number of network nodes. We have tuned the proposed GA-based algorithms and the following parameter settings are achieved: population size *pop-size* = 20, crossover probability $P_c = 0.4$ for *crossover I*, crossover probability $P_c = 0.4$ for *crossover II*, mutation probability $P_m = 0.01$ for *mutation I*, and mutation probability $P_m = 0.01$ for *mutation II*. The experiments mainly test the convergence ability, the convergence speed, and the tree cost of the achieved solutions.

Figure 5, and 6 show the percentage tree cost of BSMA [3], Sun GA-based algorithm [28], and Wang GA-based heuristic algorithm [33] in comparison with our proposed GA-based heuristic algorithm for different network sizes and different multicast group sizes. These Figures show that our proposed GA-based heuristic algorithm can result in a smaller average tree cost than the mentioned existing algorithms.

Figure 7 shows a typical example of the execution time of our proposed GA-based heuristic algorithm in comparison with the mentioned existing algorithms. This Figure shows that our proposed GA-based heuristic algorithm can result in a smaller execution time than the mentioned existing algorithms.

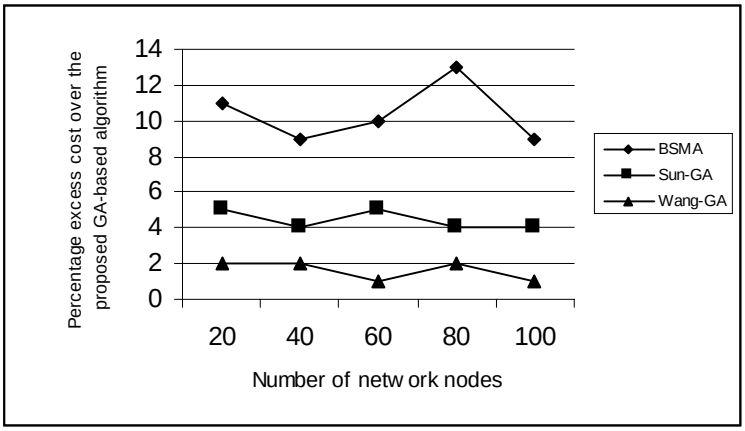

Figure 5. Percentage excess cost over the proposed GA-base algorithm versus number of network node (Multicast group size is 5% of the number of network nodes)

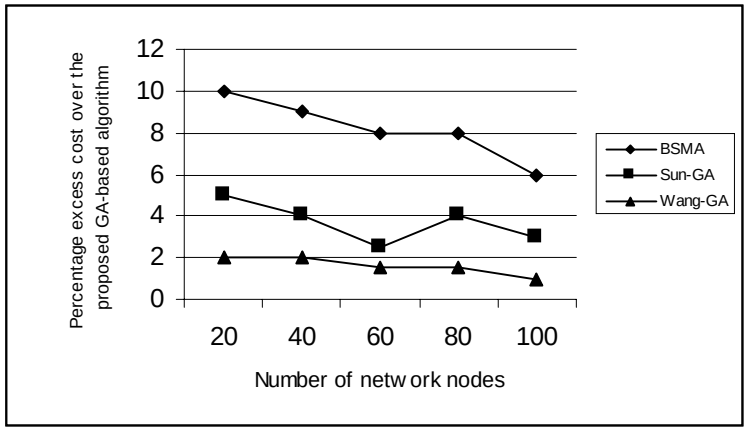

Figure 6. Percentage excess cost over the proposed GA-base algorithm versus number of network node (Multicast group size is 30% of the number of network nodes)

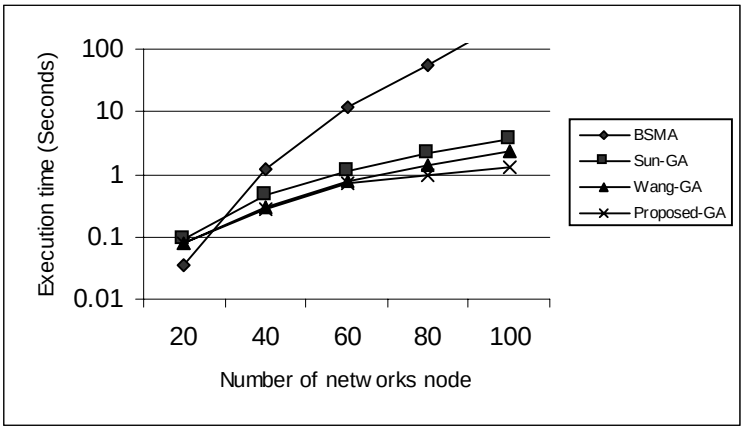

Figure 7. Execution time of the proposed algorithm in comparison with other existing algorithm

6. Conclusions

In this study, we have proposed a GA-based heuristic algorithm to solve the bandwidth-delay-constrained least-cost multicast routing problem which is known to be NP-complete. We have proposed connectivity matrix of edges for representation of the Steiner trees. In our study, the following new algorithms have been proposed to increase the performance of the genetic algorithm:

- An algorithm for creation of a random individual: *random individual creation*
- Two heuristic algorithms for mutation operator: *mutation I, II*
- Two heuristic algorithms for crossover operator: *crossover I, II*

We have used the penalizing strategy in the proposed fitness function to deal with the infeasible chromosomes and also the repairing strategy in the *mutation I* and *crossover II* algorithms to deal with the illegal chromosomes. On the other hand, we have proposed the *avoidance* strategy to avoid of creating illegal chromosomes in the *crossover I*, *mutation II*, and *random individual creation* algorithms.

We have implemented more than 3,000 lines C++ program to simulate all of the proposed algorithms. The simulation results have shown that the proposed GA-based algorithm has overcome all of the previous algorithms in the literatures.

In this study, we have focused on the source routing and the future work should focus on mechanisms to apply the proposed algorithms to the hierarchical routing.

References

[1] S. L. Hakimi, "Steiner problem in graphs and its implications," Networks, Vol. 1, pp. 113-133, 1971.

[2] R. Karp, "Reducibility among combinatorial problems," in: R. E. Miller, J. W. Thatcher, "*Complexity of computer computations*," Plenum Press, New York, pp. 85-103, 1972.

[3] M. Parsa, Q. Zhu, J.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, 1998.
[4] V.P. Kompella, J.C. Pasquale, G.C. Polyzos, "Multicast routing for multimedia communication," IEEE/ACM Transactions on Networking, Vol. 1, No. 3, pp. 286-292, 1993.
[5] R. Widyono, "The design and evaluation of routing algorithms for real-time channels," Technical Reports TR-94-024, Tenet Group, Dept. of EECS, University of California at Berkeley, 1994.
[6] A. G. Waters, "A new heuristic for ATM multicast routing," 2nd IFIP Workshop on Performance Modeling and Evaluation of ATM networks, 1994.
[7] L. Kou, G. Markowsky L. Berman, "A fast algorithm for steiner trees," Acta Informatica, Vol. 15, pp. 141-145, 1981.
[8] V. Rayward-smith, "The computation of nearly minimal steiner trees in graphs," International Journal of Mathematical Education in Science and Technology, Vol. 14, No. 1, pp. 15-23, 1983.
[9] H. Takahashi, A. Matsuyama, "An approximate solution for the Steiner problem in graphs," Mathematica Japonica, Vol. 22, No. 6, pp. 573-577, 1980.
[10] E. Gelenbe, A. Ghanwani, V. Srinivasan, "Improved neural heuristics for multicast routing," IEEE Journal of selected Area in Communication, Vol. 15, No. 2, pp. 147-155, 1997.
[11] Q. Sun, H. Langendörfer, "An efficient delay-constrained multicast routing algorithm," Journal of High-Speed Networks, Vol. 7, No. 1, pp. 43-55, 1998.
[12] L. Guo, I. Matta., "QDMR: an efficient QoS dependent multicast routing algorithm," Proceedings of the Fifth IEEE Real-Time Technology and Applications Symposium, 1999.
[13] G.N. Rouskas, I. Baldine, "Multicast routing with end-to-end delay and delay variation constraints," IEEE Journal on Selected Areas in Communications, Vol. 15, No. 3, pp. 346-356, 1997.
[14] M.V. Marathe, R. Ravi, R. Sundaram, S.S. Ravi, D.J. Rosenkrantz, H.B. Hunt , "Bicriteria network design problems," Journal of Algorithms, Vol. 28, No. 1, pp. 142-171, 1998.
[15] R. Sriram, G. Manimaran, S.R. Murthy, "Algorithms for delay-constrained low-cost multicast tree construction," Computer Communications, Vol. 21, No. 18, pp. 1693-1706, 1998.
[16] R. Sriram, G. Manimaran, S.R. Murthy, "A rearrangeable algorithm for the construction of delay-constrained dynamic multicast trees," Proceedings of the Conference on Computer Communications , IEEE INFOCOM 99, New York, March 1999.
[17] H.F. Salama, D.S. Reeves, Y. Viniotis, "Evaluation of multicast routing algorithms for real-time communication on high-speed networks," IEEE Journal on Selected Areas in Communications, Vol. 15, No. 3, pp. 332-345, 1997.
[18] P. Chotipat, C. Goutam, S. Norio, "Neural network approach to multicast routing in real-time communication networks," IEEE International Conference on Network Protocols, pp. 332-339, 1995.
[19] S. Pierre, G. Legault, "A genetic algorithm for designing distributed computer network topologies," IEEE Transactions on Systems Man and Cybernetics, Part B: Cybernetics, Vol. 28, No. 2, pp. 249-258, 1998.
[20] M.S. Bright, T. Arslan, A. Genetic, "A genetic framework for the high-level optimisation of low power VLSI DSP systems," IEEE Electronic Letters, Vol. 32, No. 13, pp. 1150-1151, 1996.
[21] Carlos A. Coello, Alan D. Christiansen, H. Aguirre Arturo, "Use of evolutionary techniques to automate the design of combinational circuits," International Journal of Smart Engineering System Design, Vol. 2, No. 4, pp. 299-314, 2000.
[22] F.K. Hwang, D.S. Richards, P. Winter, "The Steiner Tree Problem," Elsevier Science, Amsterdam, 1992.
[23] J. Hesser, R. Männer, O. Stucky, "Optimization of Steiner trees using genetic algorithms," Proceedings of the Third International Conference on Genetic Algorithms, San Mateo, CA, pp. 231-236, 1989.
[24] B.A. Julstrom, "A genetic algorithm for the rectilinear Steiner problem," Proceedings of the 5th International Conference on Genetic Algorithms, pp. 474-480, 1993.
[25] A. Kapsalis, V.J. Rayward-Smith, G.D. Smith, "Solving the graphical Steiner tree problem using genetic algorithms," Journal of the Operational Research Society, Vol. 44, No. 4, pp. 397-406, 1993.
[26] H. Esbensen, "Computing near-optimal solutions to the Steiner problem in a graph using a genetic algorithm," Networks, Vol. 26, pp. 173-185, 1995.
[27] Y. Leung, G. Li, Z.B. Xu, "A genetic algorithm for the multiple destination routing problems," IEEE Transactions on Evolutionary Computation, Vol. 2, No. 4, pp. 150-161, 1998.
[28] Q. Sun, "A genetic algorithm for delay-constrained minimum-cost multicasting," Technical Report, IBR, TU Braunschweig, Butenweg, 74/75, 38106, Braunschweig, Germany, 1999.
[29] F. Xiang, L. Junzhou, W. Jieyi, G. Guanqun, "QoS routing based on genetic algorithm," Computer Communications, Vol. 22, pp. 1394-1399, 1999.
[30] C.P. Ravikumar, R. Bajpai, "Source-based delay-bounded multicasting in multimedia networks," Computer Communications, Vol. 21, pp. 126-132, 1998.
[31] Q. Zhang, Y.W. Lenug, "An orthogonal genetic algorithm for multimedia multicast routing," IEEE Transactions on Evolutionary Computation, Vol. 3, No. 1, pp. 53-62, 1999.
[32] J. J. Wu, R. H. Hwang, H. I. Lu, "Multicast routing with multiple QoS constraints in ATM networks," Information Sciences, Vol. 124, pp. 29-57, 2000.
[33] Z. Wang, B. Shi, E. Zhao, "Bandwidth-delay-constrainted least-cost multicast routing based on heuristic genetic algorithm," Computer Communications, Vol. 24, pp. 685-692, 2001.
[34] C. Guoliang, W. Xufa, Z. Zhenquan, et al., "*Genetic Algorithm and its Application*," People's Posts and Telecommunications Press, 1996.
[35] G.N. Rouskas, I. Baldine, "Multicast routing with end-to-end delay and delay variation constraints," IEEE Journal on Selected Areas in Communications, Vol. 15, No. 3, pp. 346-356, 1997.
[36] M. Gen, R. Cheng, "*Genetic algorithms and engineering optimization*," John Wiley & Sons, 2000.
[37] G. Zhou, M. Gen, "An effective genetic algorithm approach to the quadratic minimum spanning tree problem," Computers and operations research, Vol. 25, No. 3, pp. 229-247, 1998.
[38] C. C. Palmer, "An approach to a problem in network design using genetic algorithms," Ph.D. Dissertation in Computer Science, Polytechnic University, April 1994.