# A Genetic Algorithms Based Approach for Group Multicast Routing

Luca Sanna Randaccio
Department of Electrical and Electronic Engineering of University of Cagliari, Cagliari, Italy
E-mail: L.SannaRandaccio@diee.unica.it

Luigi Atzori
Department of Electrical and Electronic Engineering of University of Cagliari, Cagliari, Italy
E-mail: L.Atzori@diee.unica.it

*Abstract*— **Whereas multicast transmission in one-to-many communications allows the operator to drastically save network resources, it also makes the routing of the traffic flows more complex then in unicast transmissions. A huge amount of possible trees have to be considered and analyzed to find the appropriate routing paths. To address this problem, we propose the use of the genetic algorithms (GA), which considerably reduce the number of solutions to be evaluated. A heuristic procedure is first used to discern a set of possible trees for each multicast session in isolation. Then, the GA are applied to find the appropriate combination of the trees to comply with the bandwidth needs of the group of multicast sessions simultaneously. The goodness of each solution is assessed by means of an expression that weights both network bandwidth allocation and one-way delay. The resulting cost function is guided by few parameters that can be easily tuned during traffic engineering operations; an appropriate setting of these parameters allows the operator to configure the desired balance between network resource utilization and provided quality of service. Simulations have been performed to compare the proposed algorithm with alternative solutions in terms of bandwidth utilization and transmission delay.**

*Index Terms*— **Group multicast routing; Multicast services; Genetic Algorithms.**

## I. INTRODUCTION

Multicast transmission of multimedia data is a crucial service provided by the network layer; in fact, it allows the operator to spare a huge amount of network resources in many circumstances. One of the most important applications that benefit form multicast transmissions is the video-clip streaming, where the same content is often sent simultaneously to even millions of users in the Internet. Another application is the IPTV, which is going to be released by most of the ISP providers in the next few years. But many others can be cited.

An important problem when implementing multicast services is the design of the multicast trees, which influences the quality and should take into account the network utilization. First works addressing this problem dealt with a single multicast session and focused on minimizing the transmission cost of each single tree. Many heuristic algorithms have been presented to solve the NP-complete unconstrained case [1]-[2], known as the Steiner tree problem. Other works, such as [3]-[5], have extended this problem by introducing constraints on the resulting quality of service (QoS), often evaluated in terms of end-to-end transmission delay.

Since in the real world several multicast sessions occur simultaneously, a new and more complex optimization problem needed to be represented: the group multicast routing problem, which consists in the study of the best combination of trees for more sessions concurrently. Until now, only few papers have been published on this topic [6]-[9]. In particular, Chen *et al.* [8] used an integer-programming approach considering only sessions with the same bandwidth requirements. They defined the multicast packing problem in which the network tried to accommodate simultaneously all the multicast groups while avoiding bottlenecks on the links with high throughput (i.e., minimized the maximum congested link shared among multicast groups). Minimization of maximum congestion is achieved at the expense of increasing the size of some multicast trees which in turn affected the delay. This trade-off was addressed by adding a penalty term to the objective function of the optimal packing formulation. The penalty term was a function of the amount of dilation from the size of the optimal tree obtained for each multicast session independently from the others, that is, in isolation. Since the mathematical programming formulation for the optimization problem was computationally intractable, they resorted to suboptimal solutions with heuristics.

---

Based on “Routing Multiple Multicast Services using Genetic Algorithms”, by L. Sanna Randaccio, L. Atzori, N. Aste which appeared in the Proceedings of the IEEE International Conference on Consumer Communications and Networking, Las Vegas, NV, January 2006.

Their heuristic method aimed to reduce the sharing of a link while ensuring that the size of multicast trees will never exceed alpha times the size of the optimum tree for a multicast group in isolation. Optimum multicast tree for each group (in isolation) was computed by using cutting-plane inequalities and the branch-and-cut algorithm. Differently, Wang *et al.* [9], considered a multicast routing problem with multiple multicast sessions under a capacity limited constraint (there is no analysis on delay). In order to solve this problem they proposed two heuristic algorithms, Steiner-tree-based and cut-set-based. If the available bandwidth for the service is just enough, these algorithms may fail to find a solution even if the solution exists. Their heuristics make use of a simple distance based cost function

In the last years, the genetic algorithms (GA) are gaining an increasing interest for solving complex problems in the networking field, as network design [10] and unicast routing [11]. GA for multicast routing without constraints was presented by Hwang *et al.* [12] and Bhattacharya *et al.*[13], while Chen *et al.* [14] and Hamdan *et al.* [15] addressed the constrained problem taking into account the QoS level provided for real-time applications in single multicast sessions.

In this paper, we address the problem of group multicast routing by exploiting the potentialities of the GA to reduce the computation complexity. The proposed solution relies on two main steps: a set of possible solutions for each session in isolation are first found; then, the combinations of these are evaluated by means of a cost function $f_C(D,B)$, which weights delay (*D*) and network resource utilization (*B*) characteristics concurrently. The cost function has been defined by identifying some combinations of delay and resource utilization that are advantageous, disadvantageous, or optimal from the operator point of view. Regions encompassing these combinations have been defined in the delay-bandwidth utilization plane (called *D-B* plane). The exact definition of the regions border is left to the operator on the basis of his needs. To this, the cost function expression has been defined in terms of few basic parameters.

The paper is organized as follows. Section 2 describes the problem of group multicast routing. Section 3 illustrates the proposed GA-based solution. Section 4 analyses the overall complexity and last section provides the experimental results.

## II. GROUP MULTICAST ROUTING PROBLEM

The problem of group multicast routing is defined as follows: given an existing network with known unicast traffic, find the optimal link capacity assignment to accommodate the multicast traffic generated by a group of multicast sources. The optimality has to be defined on the basis of the type of services conveyed through the multicast sessions and the operator objectives; yet, bandwidth usage and transmission delay are widely used in this context.

We modeled the communication network by a directed graph $G(\mathbf{V},\mathbf{E})$, where $\mathbf{V}$ is a finite set of vertices (network nodes) $n_i$ and $\mathbf{E}$ is the set of edges (network links) $l_{ij}$ with capacity $c_{ij}$ and background traffic $b_{ij}$. The number of nodes and links (i.e., the cardinalities of **V** and **E)** are *N* and *E*, respectively. Let **S** be a subset of **V** denoting the group of multicast source nodes: $\mathbf{S} \subset \mathbf{V}$, $|\mathbf{S}|=S$, and $S<N$. For each session *z,* a source node $s_z$ transmits to its multicast destination group $\mathbf{M}_z$ at rate $y_z$. The set $\mathbf{M}_z$ is included in **V** and the number of $m_{kz}$ nodes in $\mathbf{K}_z$ is $K_z$, with $z=1,...,S$. A solution to the group multicast routing problem was represented by a set of multicast trees $\mathbf{F}=\{\mathbf{T}_1,...,\mathbf{T}_S\}$, where each tree $\mathbf{T}_z$ is rooted at $s_z$ and $\mathbf{K}_z$ is the set of leaf nodes. There may be many solutions for a set of multicast sessions. The optimal set of trees $\mathbf{F}_{opt}$ depends on the objective of the operator, which are typically based on the network usage, transmission delay and target quality of service. Quite often the goodness of each possible solution is evaluated by means of a certain cost function under a given set of constraints. In that case, solving the group multicast routing problem is equivalent to find the optimal set of distribution trees on the basis of such cost function.

## III. PROPOSED SOLUTION

In our solution the benefits of the GA algorithms are used to find out an appropriate (sub-optimal) combination of multicast trees to route the data packets generated by each source $s_z$. The chromosome is then a combination of *S* genes $g_z$, each one representing a possible multicast tree for a specific session. $g_z$ is an integer number, which is indeed the index identifying a multicast tree among a given set for the multicast session *z* (*z*=1,…,*S*). This approach demands for an initial computation of possible multicast solutions for each session in isolation, which is found by combining unicast paths. The following subsections describe: the procedure to find out the sets of trees for each multicast solution in isolation, the cost function adopted to evaluate the goodness of each solution, and the application of the GA algorithm.

### *A. Multicast trees in isolation*

The potential solutions for the multicast routing problem in isolation are found by combining unicast paths connecting every source-destination couple in a session. Firstly, the paths characterized by the lowest number of hops are found out by applying the Dijkstra [17] shortest path algorithm between each source-destination couple. Secondly, alternative paths are computed by modifying the shortest paths. These are

```
Algorithm SearchAdditionalPaths;
  Input:  network of N nodes
          vectors of shortest paths $\mathbf{D}_{i,j}$, for $i,j = 1, ..., N$;
          source node $s_z$;
          destination node $m_{kz}$;
  Output: P additional paths $AP_p$, for $p=1,…,P$
  begin
    $l = |\mathbf{D}_{s_z,m_{kz}}|$;
    $p = 1$;
    while ($l > 1$) do
      begin
        $l = l - 1$;
        $\Psi$ := set of nodes adjacent to $D_{s_z,m_{kz}}(l)$;
        for any $\varphi \in \Psi$ and $\varphi \not\subset \mathbf{D}_{s_z,m_{kz}}$ do
          begin
            $AP_p = \mathbf{D}_{s_z,\varphi} \cup \mathbf{D}_{\varphi,m_{kz}}$;
            $p = p+1$;
          end;
      end;
```

Fig. 1. Pseudocode for the heuristic aimed at finding the additional paths.

named *additional paths* in the following. Since this procedure needs shortest paths also for nodes not representing source-destination pairs, the Dijkstra algorithm is applied for every couple of nodes in the networks since the beginning.

Many techniques proposed in the literature adopt either the DFS (Depth First Search) algorithm or the DFS search with a threshold in depth [11]-[16]; differently, we propose a heuristic aimed at reducing the amount of memory requirements with respect to the DFS-based solutions. Additional paths are found analyzing the nodes adjacent to those belonging to the shortest path. Let $\mathbf{D}_{i,j}$ be the shortest path connecting $n_i$ to $n_j$: it is the vector of indexes of the involved nodes. This path is truncated at position $x$, with $x$ in the range $1 \div |\mathbf{D}_{i,j}|$ -1. From each vector truncated at position $x$, we consider each node adjacent to $D_{i,j}(x)$ and the shortest path from this adjacent node to the destination. This vector is then joined with the truncated vector to form an additional path. The total number of additional paths depends on the length of the shortest path $\mathbf{D}_{i,j}$ and on the degree of adjacency. The rational behind this heuristic is twofold: finding alternative paths not too much longer than the shortest one; linking the size of this set with the size of the network. Indeed, the number of additional paths can be estimated as follows. Let $a$ and $b$ be the minimum (Dijkstra shortest path) number of hops from the source to the destination and the average degree of adjacency, respectively. An estimation of the number of additional paths is given by $(a$-1$)(b$-1$)$. Note that the additional paths are computed for each couple $s_z$-$m_{kz}$ starting with vector $\mathbf{D}_{s_z,m_{kz}}$ ($z$=1,…,$S$ and $k$=1,…,$K_z$). The pseudocode for this procedure is provided in Fig. 1. Fig. 2 shows the application of this procedure to a simple test network, where $n_1$ is the source ($s_z$) and $n_4$ and $n_5$ are two destinations ($m_{11}$ a and $m_{21}$). The graph in (a) shows the Dijkstra path from the source to destination $n_4$ with nodes and links in straight lines, i.e., $n_1$-$n_2$-$n_3$-$n_4$. Note that there are two destinations for this multicast session, $n_4$ and $n_5$, but we are considering one source-destination path at a time. Two additional paths are found by considering the nodes adjacent to node $n_3$, as shown in (b) and (c). This procedure is repeated for all the nodes belonging to the path till the source node $n_1$, bringing to the additional path shown in (d) where the path $n_1$–$n_7$–$n_5$-$n_4$ is found.

For every pair $s_z$-$m_{kz}$, let $\mathbf{L}_{kz}$ represent the two-dimensional array of the shortest and additional paths. Each row of $\mathbf{L}_{kz}$ refers to a different unicast path providing the list of relevant nodes. The first row is always the shortest path. The additional rows are filled while performing the heurist that computes the additional paths. For a specific source $z$, all the arrays $\mathbf{L}_{kz}$ ($k$=1,…, $K_z$) are then used to build the possible trees for this multicast session. These trees are extracted by grouping one-by-one $K_z$ paths taken from the $\mathbf{L}_{kz}$ arrays and deleting all the duplicate links. Information on these combinations are then stored in a two-dimensional array $\mathbf{L}_z$ of unicast paths. Each row of $\mathbf{L}_z$ is a vector of length $K_z$, which represents a possible tree for the multicast session. Indeed, each element in a row indexes a row in an array $\mathbf{L}_{kz}$ providing the information on the relevant unicast path to be considered to connect source $s_z$ to destination $m_k$. The resulting number of rows in $\mathbf{L}_z$ has been experimentally determined to be lower than $H_N \cdot \ln N$, where $H_N$ is the maximum out-degree of the network. With arrays $\mathbf{L}_z$

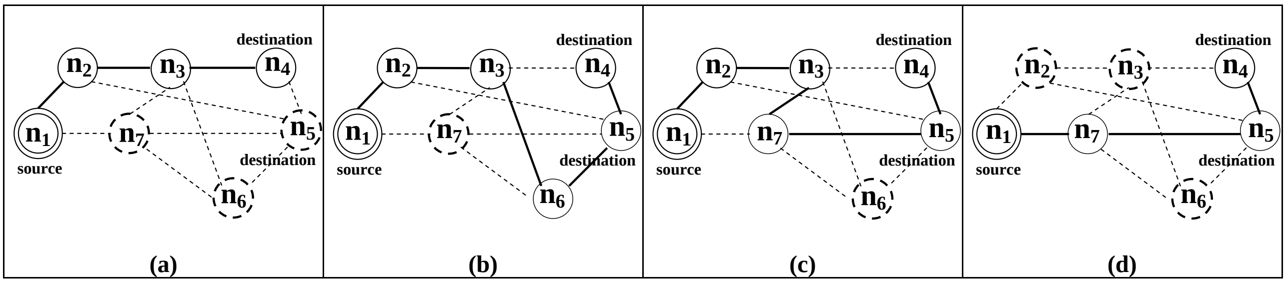

Fig. 2. Additional paths: (a) shortest path for nodes $n_1$-$n_4$; two (b) and (c) additional paths found by using nodes by replacing the last link with alternative paths; (d) additional path obtained by replacing the last two links with an alternative path.

($z$=1,…,$S$), we are then able to build the chromosome used in the GA-based algorithm. The chromosome is created by combining $S$ genes. Each gene $z$ indexes the row of array $\mathbf{L}_z$, referring to a specific multicast tree for that session.

The proposed notation is clarified with the simple example in Fig. 3. Assume that there are five multicast flows; accordingly, the chromosome is made of five genes, each one indicating a specific multicast tree. With respect to Figs. (a) and (b), the second gene $g_2$ refers to the second session ($z$=2) and is set to 6, i.e., it refers to the sixth row of array $\mathbf{L}_2$, which provides all the possible multicast trees for this session. In this example, the rows of $\mathbf{L}_2$ are of length three, meaning that in session 2 there are only three destinations. The values of the sixth row of this array provide the indexes of the rows of the arrays $\mathbf{L}_{kz}$ ($z$=2, $k$=1,2,3), which code the three unicast paths to the destinations. In this example, the three paths are the followings: $n_2$-$n_3$-$n_4$, $n_2$-$n_3$-$n_4$-$n_5$, and $n_2$-$n_3$-$n_6$.

### B. Application of the GA algorithms

As in many other applications of the GA algorithms proposed in the literature ([11]-[16]), the population is randomly initialized, creating a population $P$ of $P_{Dim}$ chromosomes, each made by $S$ genes. Note that the solution was found by making the population evolving by means of crossover and mutation operators. In particular, the most fitting part of the population $P_{best}$ was selected and directly inserted in the new generation, while the rest of the population $P_{worst}$ was discarded and replaced by a sub-population created by means of the crossover and mutation operators. A crossover operator (with probability $CROSS$) is used to interchange the elements of two strings, while mutation operator (with probability $MUT$) tries to lead the search out of local optima. In the case of two identical chromosomes resulted after the crossover and mutation operations, two individuals are randomly generated. The termination condition is satisfied once either $g$ reaches a selected number of iterations ($IT$) or the fitness function maintains the same value for $IT_{MAX}$ iterations.

### C. Evaluation

The chromosome fitness was evaluated by either functions $f_B$ or $f_C$. The first was used when the optimality is correlated with a uniform distribution of the traffic over the entire network. The second was used when the optimality depends on both network resource occupation and transmission delay. The exact definition of these cost functions is provided in the following sub-sections.

### C.1. Bandwidth contribution

The proposed cost function $f_B$ was intended to evaluate the bandwidth distribution over the network links so as to obtain the most uniform resource utilization. In this way, broad trees with low average link utilization could be selected even if there are smaller trees, but with higher average link utilization. Additionally, a constraint on the overall transmission delay for each path in each multicast session was enforced.

For a given solution $\mathbf{F}$, let $B$ represent the bandwidth contribution value computed as the average link occupation:

$$B = \frac{1}{L}\sum_{i=1}^{N}\sum_{j=1}^{N}\frac{A(y_{ij},c_{ij})}{c_{ij}}, \qquad (1)$$

where $y_{ij}$ represents the bandwidth allocation obtained summing the multicast transmission rate to the background traffic occupancy in the $l_{ij}$ link:

$$y_{ij} = b_{ij} + \sum_{z=1}^{S} x_{zij}\cdot y_z \text{ , with } \begin{cases} x_{zij}=1 & \text{if } l_{ij}\in\mathbf{T}_z \\ x_{zij}=0 & \text{otherwise} \end{cases} .(2)$$

If this allocation exceeds a threshold value $B_{Max}$

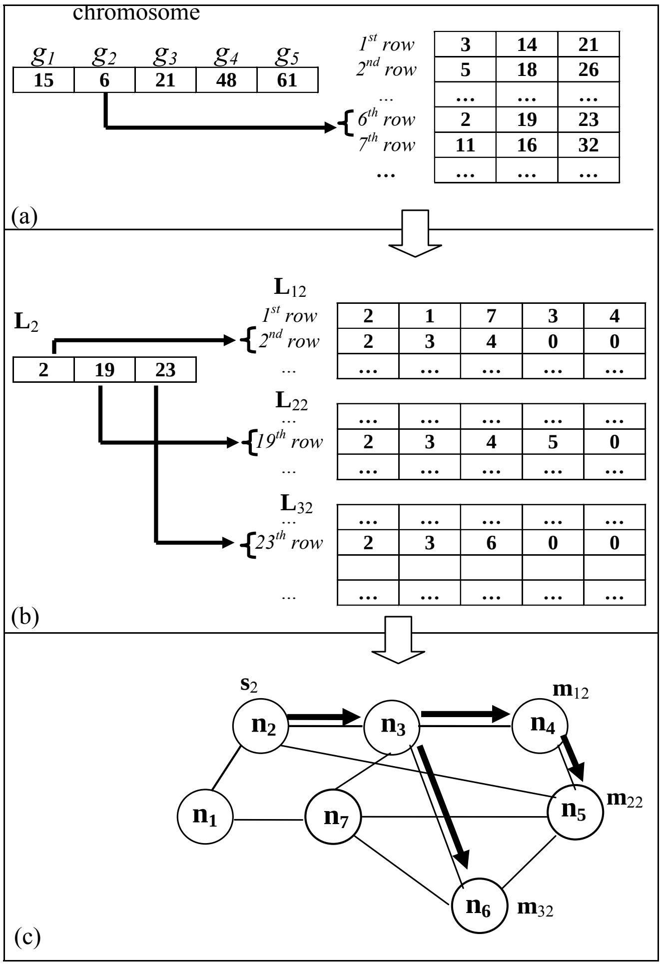

Fig. 3. Example of genotype coding. (a): chromosome coding and array $L_2$ coding the set of multicast trees for session 2; (b): second row of $L_2$ and arrays $L_{k2}$ ($k$=1,2,3) providing the unicast trees; (c): tree considered for session 2 in the chromosome in (a).

(default value equal to 90%), the link is treated as in congestion state:

$$A(a,b)=\begin{cases} a & \text{if } a<0.9\cdot b \\ b & \text{otherwise} \end{cases}. \qquad (3)$$

$L$ is the number of links involved in session routing:

$$L=\sum_{i=1}^{N}\sum_{j=1}^{N} x_{ij} \text{ ,with } \begin{cases} x_{ij}=1 & \text{if } l_{ij}\subset \mathbf{F} \\ x_{ij}=0 & \text{otherwise} \end{cases}. \qquad (4)$$

The cost function $f_B$ we proposed is exactly equal to the expression of $B$ in (1).

The delay constraint is the following:

$$d_{kz} \le D_{MAX}\,, \qquad (5)$$

that is, for each unicast path the delay has to be less than the threshold $D_{MAX}$ (e.g., 200 msec is commonly used value for streaming applications). The one-way delay $d_{kz}$ from source $z$ to destination $k$ is computed as the sum of delays $d_{ij}$ in each hop $ij$ involved in the unicast path. We used the link delay computation proposed in [18]:

$$d_{ij}=\frac{MTU}{c_{ij}-b_{ij}}+\frac{MTU}{c_{ij}}+\pi_{ij}\,. \qquad (6)$$

It is based on the sum of three terms: the expected best packet scheduling delay using Weighted Fair Queuing discipline with a maximum packet length equal to the *MTU*; the packet transmission time through the output link; and the link propagation delay $\pi_{ij}$.

### *C.2. Bandwidth and delay contribution: D-B plane*

The aim of the second fitness function $f_C$ was to weight both bandwidth usage and one-way transmission delay. These two components were combined together in a cost function that allowed the operator to obtain the “optimal” compromise between service delay and network resource usage. Clearly, there is not a universal definition of optimality in this framework because this depends on the network operator point of view. Accordingly, herein we defined a parametric cost function that provided a simple way to combine delay and network usage by setting a few parameters.

The bandwidth component used in cost function is the same used in previous subsection in (1). As to the delay component, we computed the average value among the multicast tree delays:

$$D=\left(\sum_{z=1}^{S}\alpha_z\cdot D_z\right)\Bigg/\left(D_{MAX}\cdot\sum_{z=1}^{S}\alpha_z\right), \qquad (7)$$

where the multicast tree delay $D_z$ is calculated as the average delay among the unicast paths that constitute the tree for the session $z$:

$$D_z=\frac{1}{K_z}\cdot\sum_{k=1}^{K_z} d_{kz}\,. \qquad (8)$$

In order to differentiate real-time applications from simple data transfer ones, weights $\alpha_z$ are introduced.

As previously mentioned, the transmission delay and the bandwidth usage were combined in a single cost function $f_C(D,B)$ that provides the goodness of each solution. Such a function is described by defining some particular areas in the bi-dimensional *D-B* plane. Note that both $D$ and $B$ are normalized in the range 0÷1. These areas correspond to combinations of delay and capacity usage that are either particularly advantageous, disadvantageous, or quite common for the network operator. The first encompasses low $D$ and $B$ values and represents situations of very low network resource utilization and low delays. It is highlighted in grey color in Fig. 4 and $f_C=0$ is assigned for every point within. It is named *good region* in the following for presentation convenience. The opposite situation corresponds to the “border” area of the *D-B* plane, where the capacity usage reaches high values and/or the delay is very high. This is the dashed area in Fig. 4, named *bad region*, to which a value of $f_C=1$ is assigned. Besides these two areas, we have defined a series of combinations of $D$ and $B$ that represent the most frequent operational points for the network and are considered of equal importance for the operator: $f_C$ is set to 0.5 in these points; these combinations are defined by means of a parabolic curve, named *control curve*, determined by the operator defining three points (in terms of *D-B* coordinates values); these control points are represented by the cross symbols in Fig. 4. The defined three areas constrain the shape of $f_C(D,B)$. Additionally, it is reasonable to suppose the cost function convexly increases from the good region border to the control curve and concavely increases from the control curve to the bad region border. Three curves characterized by such a behavior are depicted in Fig. 5. In particular, these represent potential curves obtained from the intersection of the bi-dimensional cost function

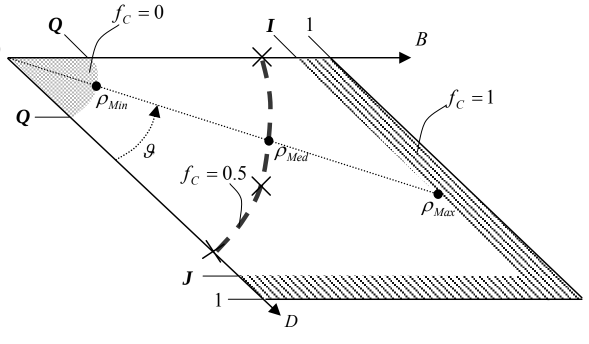

Fig. 4. *D-B* plane: grey area represents the good region; dashed area is the bad region; and dashed line is the control curve.

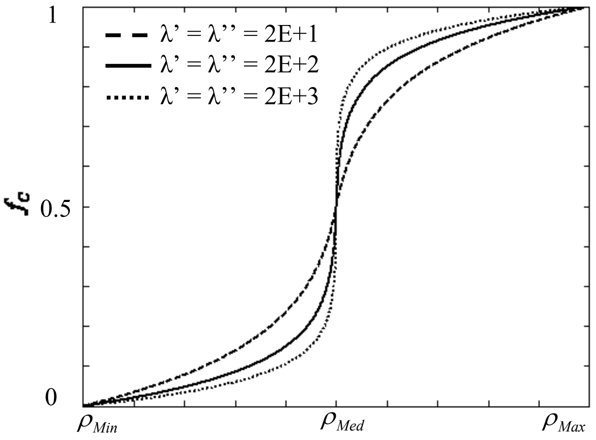

Fig. 5. Cost function behavior for a given angle for three different values of $\lambda'$ and $\lambda''$.

with a plane orthogonal to the *D-B* plane and crossing its origin.

Once outlined the main characteristics of the cost function, we now need to define its analytical expression. In particular, it has to be characterized by some parameters to be set by the operator and related to the previously introduced areas. In the following, we propose a possible solution.

As to the control curve, it is given by the following expression:

$$B = \alpha D^2 + \beta D + \chi , \qquad (9)$$

where $\alpha$, $\beta$, and $\chi$ are easily determined by enforcing the curve to cross the three control points.

The good and bad areas are defined by means of the following inequalities:

$$\text{bad area:} \qquad D \geq J, \forall B; \quad B \geq I, \forall D ,$$

$$\text{good area:} \qquad D \leq Q, \quad B \leq Q \sin(\arccos D/Q),$$

where *J* and *I* are two parameters defined by the operator and depicted in Fig. 4.

To simplify the tractability, we now move to the polar representation of the *D-B* plane, in terms of the angle $\vartheta$ and radius $\rho$ :

$$\vartheta = \arctan B/D; \quad \rho = \sqrt{B^2 + D^2} . \qquad (10)$$

At a given angle $\vartheta$, the function is desired to have the shape drawn in Fig. 5 as a function of the radius. With reference to this figure, $\rho_{Med}$ represents the value of $\rho$ on the control curve, $\rho_{Min}$ the value of $\rho$ on the good region border, and $\rho_{Max}$ the value of $\rho$ in the bad region border. Accordingly, for a given value of the angle, the function has the following expression:

$$f_C = \begin{cases} \gamma' + \kappa' \log_{10}[\lambda'(\rho - \rho_{Med}) + 1], & \rho_{Min} < \rho < \rho_{Med} \\ \gamma'' + \kappa'' \log_{10}[\lambda''(\rho - \rho_{Med}) + 1], & \rho_{Med} < \rho < \rho_{Max} \end{cases} , \qquad (11)$$

where $\lambda'$ and $\lambda''$ drive the exact shape of the cost functions. These allow the operator to either force the solution to stay close to the control curve (parameters set to high values) or tolerate solution far from this (parameters set to low values). In Fig. 5, the continuous line corresponds to $\lambda' = \lambda'' = 200$, which represents the default setting. The other four parameters in (11) are set by enforcing the constraints on the borders:

$$\begin{cases} \gamma' = 0.5 \\ \gamma' + \kappa' \log_{10}[\lambda'(\rho_{Min} - \rho_{Med}) + 1] = 0 \end{cases} \\ \begin{cases} \gamma'' = 0.5 \\ \gamma'' + \kappa'' \log_{10}[\lambda''(\rho_{Max} - \rho_{Med}) + 1] = 0 \end{cases} . \qquad (12)$$

Expression (11) is used to evaluate the cost function for a combination *D-B* of delay and capacity usage. After computing the angle $\vartheta$ by means of (10), $\rho_{Med}$ comes out from the conversion of (9) in the polar plane:

$$\rho_{Med} = \frac{\sin\vartheta - \beta\cos\vartheta \pm \sqrt{(\beta\cos\vartheta - \sin\vartheta)^2 - 4\alpha\chi\cos^2\vartheta}}{2\alpha\cos^2\vartheta} . \qquad (13)$$

If $\rho$ is higher than this value, $\rho_{Max}$ has to be computed as follows:

$$\rho_{Max} = \begin{cases} \sqrt{B^2 + J^2}, & 0 \leq \vartheta \leq \pi/4 \\ \sqrt{D^2 + I^2}, & \pi/4 \leq \vartheta \leq \pi/2 \end{cases} .$$

Differently, if $\rho < \rho_{Med}$ we don't have to compute $\rho_{Min}$, which is always equal to *Q*.

It is important to underline that (11) is not dependant on how we describe the two regions and the curve, but only on the value of $\rho$ in the area border and the curve.

## IV. COMPLEXITY ANALYSIS

The resulting time complexity does not depend on the used cost function and is expressed by: $O(Dijk + Path + Tree + Gen)$, where:

$$\begin{cases} O(Dijk) = O(N^3) \\ O(Path) = O(S \cdot K_z{}^2 \cdot N) \\ O(Tree) = O(S \cdot (H_N \cdot \ln N)^{K_z}) \\ O(Gen) = O(IT \cdot S \cdot K_z \cdot P_{\text{dim}} \cdot N^2) \end{cases} .$$

In particular, $O(Dijk)$ is the complexity of the Dijkstra algorithm, $O(N^2)$, applied to all $N$ nodes. $O(Path)$ represents the complexity for generating additional paths: in fact, all $K_z$ destination nodes need to be analyzed for all *S* sessions and new paths are searched starting from the best ones previously found. $O(Tree)$ is the complexity for the tree generation procedure: the maximum number of paths found for each destination node is $H_N \cdot \ln N$, i.e., the upper bound for the number of $\mathbf{L}_z$ arrays; this procedure is iterated for all $K_z$ nodes and has to be repeated for all $S$ sessions. $O(Gen)$ represents the genetic algorithm complexity, considering $N \times N$ matrices and $K_z$ destination nodes for all $S$ sessions.

Even though the complexity quickly increases as network and destination nodes number grows, it has to be highlighted that these are intended as routers, and not as end-stations. For this reason, high $N$ values (i.e. more than 50) and $K_z$ values (i.e. more than 20) are very uncommon to be reached.

## V. EXPERIMENTS

### A. Settings

In this section, we present the results for different network configurations. The network topologies used in the simulations were generated by a random graph model proposed by Waxman [19]. The generator first randomly distributes $N$ nodes over a square coordinate grid. The link between any two nodes $n_i$ and $n_j$ is added by the probability function $P((n_i, n_j)) = \beta \exp(-d(n_i, n_j)/\alpha \cdot l)$, where $d(n_i, n_j)$ is the Cartesian distance between nodes $n_j$ and $n_j$, $l$ is the maximum possible distance between any two nodes, and parameters $\alpha$ and $\beta$ are real numbers in the range (0,1]. Note that these parameters can be appropriately tuned to obtain the desired characteristics in the resulting graph. In our simulations, we set $\alpha = 0.4$ and $\beta = 0.5$ for $N$=20 nodes. We used connected graphs only: if generated graph are not connected, it is discarded.

In the following, we refer to the GA algorithm using cost functions $f_B$ and $f_C$ with GA-B and GA-C, respectively. Our results were compared with the solutions calculated using the cost functions proposed by Wang [9] and Chen [8]. It has to be stressed that we calculate them by using genetic algorithms and not by their heuristics: thus we referred to them as GA-Wang and GA-Chen, respectively. This choice has been made since the link cost adopted by Chen and Wang have a different meaning with respect to the cost function proposed in our paper, thus a comparison in terms of resulting minimum cost and processing time is not appropriate; instead, we think it is important to compare the traffic distribution in the network, in terms of bandwidth and delay. As proposed by Wang, the link cost was set to be the distance, obtained using the Waxman model, between two end-nodes of that link (and normalized when running the Chen algorithm), while the capacity constraint for each link was assigned to be the link cost times a random number in [2/$N$, 1].

The member node sets are the same for the sessions (i.e., $\mathbf{M}_1 = \mathbf{M}_2 = ... = \mathbf{M}_z = \mathbf{M}$) and the number of multicast sessions is equal to the size of member node set (i.e., $S = |\mathbf{M}|$). This has been done to have results directly comparable with the ones of the Wang's algorithm.

The control curve has been set with the following *D-B*

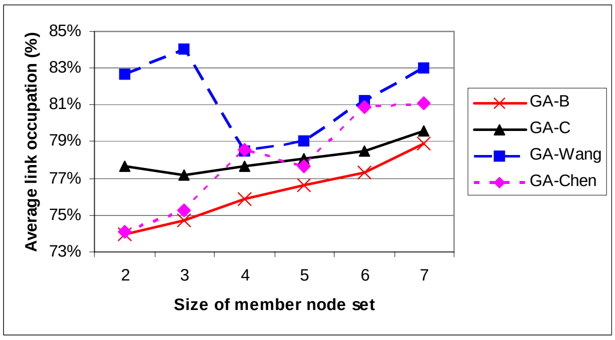

Fig. 6. Multicast links occupation.

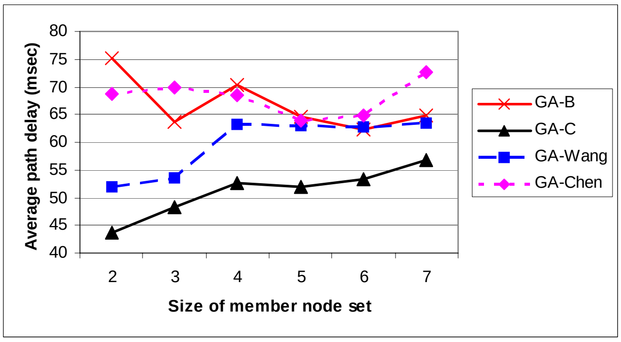

Fig. 7. End-to-End delay.

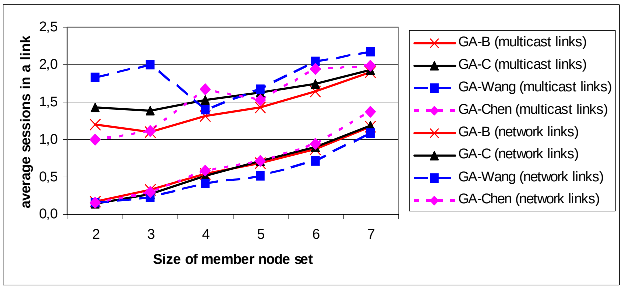

Fig.8. Number of sessions per links.

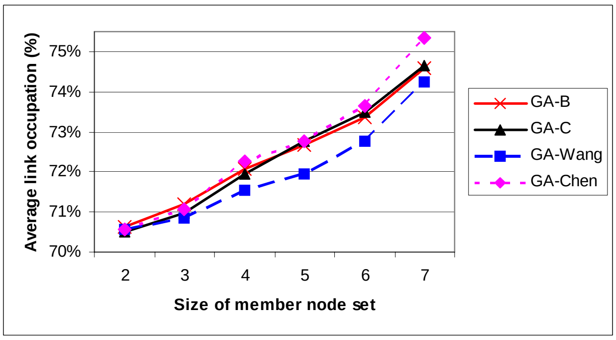

Fig. 9. Network links occupation.

control points: (0.8, 0), (0.5, 0.4) and (0, 0.8); the good and bad areas borders are set with $I = J = 0.9$ and $Q = 0.2$; the cost function behavior is the default one with $\lambda' = \lambda'' = 200$.

### B. Test analysis

At first we have run some tests to analyze the importance of the GA parameters: $P_{Dim}$ was varied in the range (8, 16, 32, 64), $CROSS$ and $MUT$ in the range 0.3÷0.7 and 0.01÷0.3, respectively. The maximum difference in terms of final cost function value among all the solutions was observed to be less than 2%, which may be tolerated in most of cases. The following considerations can be made: a high population size brings to better solutions at the expense of a higher processing time; the $MUT$ parameter is suggested to be set equal to or higher than 0.1 to avoid an excessive number of iterations; the $CROSS$ parameter does not sort significant effects in the range used. For these reasons, in the following experiments the adopted parameter values are: $CROSS$=50%, $MUT$=30%, $P_{Dim}$=32, $IT$=1000, and $IT_{MAX}$=200.

To observe the behavior of the proposed algorithms in critical conditions, the network was loaded with background traffic close to 70%. In particular, the background unicast traffic occupation for each link was randomly set in the range 50%÷80%, while the link propagation delay $\pi_{ij}$ was randomly set in the range 1÷40 msec for each link. When applying GA-C, all flows have been treated evenly ($\alpha_z = 1$ and $y_z = 1$ for every $z$).

Figs. 6-9 show the results when the number of multicast sessions is set in the range 2÷7. Recall that $K_z$=$S$ for every $z$=1,…,$S$. As shown in Fig. , the GA-B algorithm obtained the best results in terms of bandwidth occupation in the links used by the multicast sessions at the expense of higher end-to-end delays, as shown in Fig. 8, where GA-B have low values, but not the best ones. The GA-C algorithm brought to a higher bandwidth occupation with respect to GA-B, but it provided the best results in the end-to-end delays. As to the Wang algorithm, it reached its scope using the links with the lowest weight, not necessarily linked to the bandwidth, and was not interested in optimizing this bandwidth usage as shown in Fig. 6. However, it provided the lowest end-to-end delay in some circumstances (see Fig. 7). As to the Chen algorithm, it was interested in reducing the traffic on the most loaded links, thus leading to a better usage of the link occupancy; however, it didn't take into account the delays, bringing to the worst results. It has to be noted that the proposed algorithms allowed for an optimal usage of the multicast links but led to a growth of the resulting trees in terms of number of hops involved, as shown in Fig. 8 (upper set of curves): the number of sessions per used link with GA-B and GA-C was lower than the ones for the GA-Wang and the GA-Chen, which tend to pack the traffic flows. However, this figure is greater when considering the sessions over all the network links (lower set of curves in Fig. 8) because GA-B and GA-C used more links; consequently, the total traffic generated was greater too. However, this traffic growth is not very significant: as Fig. 9 shows, the proposed algorithms create less than 1% traffic more than GA-Wang algorithm, while have comparable to the GA-Chen algorithm. Referring to proposed algorithms, this spanning phenomenon is obviously lower with the GA-C algorithm, since many hops lead to high delays.

We have also analyzed the results obtained when using the DFS algorithm during the path search phase to build the space solution instead of the Dijkstra and the proposed heuristic for additional path computation. We refer to this algorithm with GA-DFS, whose results are compared with the GA-B and GA-C ones and show in Table 1. In particular, we provide the difference normalized by the GA-DFS results. The end-to-end delay, multicast link occupation, network link occupation, cost and processing time have been considered in this analysis. It has to be noted that the final costs obtained with the proposed algorithms were only slightly higher than GA-DFS, with a normalized differences at most of 0.4%, in spite of the significant processing time growth. As to the delay and link occupation, it can be noted that the gap above 4% are always negative, i.e., occurred when our algorithms performed better (even if not in terms of final costs). It is important to underline that the results for more than five sources are not shown since the memory required to perform the GA-DFS was too high.

TABLE I.
COMPARISON OF THE PROPOSED GENETIC ALGORITHMS GA-B AND GA-C WITH THE GA-DFS, WHERE THE DFS ALGORITHM IS USED INSTEAD OF THE DIJKSTRA, AND THE PROPOSED HEURISTIC FOR ADDITIONAL PATH COMPUTATION. THESE RESULTS ARE RELEVANT TO NORMALIZED DIFFERENCE BETWEEN THE PROPOSED ALGORITHM AND THE GA-DFS ONE.

| Number of sources | End-to-End delay | | Multicast link occupation | | Network link occupation | | Final cost | | Processing time | |
|---|---|---|---|---|---|---|---|---|---|---|
| | GA-B | GA-C | GA-B | GA-C | GA-B | GA-C | GA-B | GA-C | GA-B | GA-C |
| 2 | 0.00% | -6.91% | 0.00% | 3.50% | 0.00% | -0.08% | 0.00% | 0.40% | -8.55% | -15.02% |
| 3 | -18.47% | -2.90% | 0.43% | 1.23% | 0.00% | -0.16% | 0.46% | 0.05% | -31.52% | -14.75% |
| 4 | -10.08% | 1.57% | 0.44% | 0.71% | 0.00% | 0.16% | 0.47% | 0.22% | -46.68% | -41.68% |
| 5 | -20.55% | 2.13% | 0.39% | 0.00% | -0.16% | 0.08% | 0.10% | 0.20% | -75.06% | -80.86% |

## VI. CONCLUSIONS

In this paper we presented the application of the genetic algorithm to reduce the computational complexity in group multicast routing. Additionally, a novel cost function was proposed which weights network bandwidth allocation and quality of service, in term of one-way delay. The expression of the cost function has been devised so as to allow the operator to configure the desired balance between network resource utilization and provided quality of service by tuning few parameters. Experimental results showed that the proposed algorithm allows obtaining a better distribution of the session traffic, avoiding the creation of undesired bottlenecks.

## REFERENCES

[1] A. Shaikh, S. Lu, and K. Shin, "Localized multicast routing", *Proceedings of IEEE Globecom'95*, pp. 1352-1356, 1996.

[2] H. Takahashi and A. Matsuyama, "An approximation solution for the Steiner tree problem in graphs", *Mathematica Japonica*, vol. 24, pp. 573-577, 1980.

[3] X. Jia, "A distributed algorithm of delay-bounded multicast routing for multimedia applications in wide area networks", *IEEE/ACM Transaction on Networking*, vol. 6, pp. 828-837, 1995.

[4] Q. Zhu, M. Parsa, and J. Garcia-Luna-Aceves, "A source-based algorithm for delay-constrained minimum-cost multicasting", *Proceedings of IEEE Infocom'95*, pp. 377-385, 1995.

[5] C. Chung, S. Hong, and H. Huh, "A fast multicast routing algorithm for delay-sensitive applications", *Proceedings of Globecom'97*, pp. 1898-1902, 1997.

[6] C. Wang, B. Lai, R. Jan, "Optimum multicast of multimedia streams", *Computer & Operations Research*, vol. 26, pp. 461-480, 1999.

[7] V. Priwan, H. Aida, and T. Saito "The multicast tree based routing for the complete broadcast multipoint-to-multipoint communications", *IEICE Transactions of the Communications*, vol. E78-B, pp.720-728, 1995.

[8] S. Chen, O. Günlük, and B. Yener, "The multicast packing problem", *IEEE/ACM Transactions on Networking*, vol. 8, pp. 311-318, June 2000.

[9] C. Wang, C. Liang, and R. Jan, "Heuristic algorithms for packing of multiple-group multicasting", *Computer & Operations Research*, vol. 29, pp. 905-924, 2002.

[10] L. Atzori and A. Raccis, "Network Capacity Assignment for Multicast Services Using Genetic Algorithms", *IEEE Communications Letters*, vol. 8, no. 6, pp. 403-405, June 2004.

[11] C.W. Ahn and R.S. Ramakrishna, "A genetic algorithm for shortest path routing problem and the sizing of population", *IEEE Transaction on evolutionary computation*, vol. 6, no. 6, Dec. 2002.

[12] R.H. Hwang, W.Y. Do, and S.C. Yang, "Multicast Routing Based on Genetic Algorithms", *Journal of information science and engineering*, vo1.16, pp.885-901, 2000.

[13] R. Bhattacharya, P. Venkateswaran, S.K. Sanyal, R. Nandi, , "Genetic algorithm based efficient routing scheme for multicast networks", *IEEE International Conference on Personal Wireless Communications (ICPWC 2005)*, pp. 500 - 504, Jan 2005.

[14] L. Chen, Z. Yang, Z. Xu, "A degree-delay-constrained genetic algorithm for multicast routing tree", *The Fourth International Conference on Computer and Information Technology (CIT)*, pp. 1033 - 1038, September 2004.

[15] M. Hamdan and M.E. El-Hawary, "Multicast routing with delay and delay variation constraints using genetic algorithm", *Canadian Conference on Electrical and Computer Engineering*, vol. 4, pp. 2363 - 2366, May 2004.

[16] A.T. Haghighat, K. Faez, M. Dehghan, A. Mowlaei and Y. Ghahremani, "GA-based heuristic algorithms for bandwidth-delay-constrained least cost multicast routing", *Computer Communications*, vol. 27, pp. 111-127, 2004.

[17] E. W. Dijkstra "A note on two problem in connexion with graphs", *Numerische Mathematik*, vol.1, pp. 269–271, 1959.

[18] J. Zhao, H. Hassanein, J. Wu and J. Luo, "CRMA: A Cycle-breaking Multicast Routing Algorithm for Supporting QoS over the Internet", *Proceedings of the 36th Hawaii International Conference on System Sciences*, 2003.

[19] B. Waxman, "Routing of multipoint connections", *IEEE Journal on Selected Areas in Communications*, vol. 6, pp. 1617-1622, December 1988.

**Luca Sanna Randaccio** received the Laurea (MSc) degree in Electronic Engineering in 2002, from the University of Cagliari, Italy, and his PhD degree in Electronic Engineering and Computer Science from the University of Cagliari in 2005. He actually collaborates with the Department of Electrical and Electronic Engineering of Cagliari as a PostDoc Fellow in the telecommunication field.

Since 2001 he is collaborating on research activities regarding next generation networks; in particular, after collaborating at projects regarding multi-service networks based on ATM switching technology with the Ericsson Research Center located in Cagliari, he is now involved in research projects on multi-service multi-domain and multi-vendor interoperability in NGN networks based on IP technology.

Dr. Sanna Randaccio is student member of IEEE since 2005.

**Luigi Atzori** (M'97) is an assistant professor in telecommunications at the University of Cagliari, Italy.

He received the PhD degree in Electronic Engineering and Computer Science from the University of Cagliari in 1999. In 1994 he spent four months as a visiting student at the Technical University in Braunschweig (Germany) and in 1996 he received the Telecom Italia award for an outstanding M.Sc. thesis on multimedia networking. He has been awarded a Fulbright Scholarship (2003-2004) to work on video streaming at the Department of Electrical and Computer Engineering, University of Arizona. His main research topics of interest are in the multimedia signal processing and transmission: error recovery and concealment, video post-processing, IP telephony playout buffering, video streaming. He is also interested in data network performance analysis. He has published more than 45 journal articles and refereed conference papers.

Dr. Atzori is member of IEEE since 1996 and member of the IEEE Multimedia Communications Committee (MMTC) since 2005. He serves on the organizing and technical committees of leading multimedia communications conferences.