Bandwidth Constrained IP Multicast Traffic Engineering Without MPLS Overlay

Ning Wang and George Pavlou

Center for Communication Systems Research, University of Surrey, United Kingdom
{N.Wang, G.Pavlou}@surrey.ac.uk

Abstract. Existing multicast traffic engineering (TE) solutions tend to use explicit routing through MPLS tunnels. In this paper we shift away from this overlay approach and address the bandwidth constrained IP multicast TE directly based on link state IGP routing protocols. The objective is that, through plain PIM-SM shortest path routing with optimized Multi-topology IS-IS (M-ISIS) link weights, the resulting multicast trees are geared towards minimal consumption of bandwidth resources. We apply Genetic Algorithms (GA) to the calculation of optimized M-ISIS link weights that specifically cater for engineered PIM-SM routing with bandwidth guarantees. Our evaluation results show that GA-based multicast traffic engineering consumes significantly less bandwidth resources in comparison with conventional IP approaches, while it also exhibits higher capability of eliminating/alleviating link congestion. The key contribution is a methodology for engineering multicast flows in a pure IP environment, without MPLS explicit routing that potentially suffers from scalability problems in terms of LSP maintenance.

1 Introduction

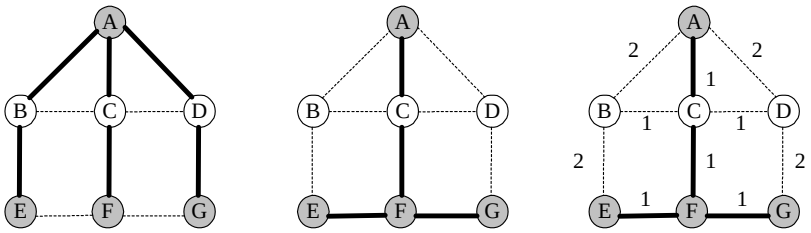
Traffic Engineering (TE) [1] is an efficient mechanism for improving the service capability of operational IP networks. In literature, traffic engineering approaches can be classified into Multi-Protocol Label Switching (MPLS) based and pure IP-based. With MPLS-based TE, packets are encapsulated with labels at ingress points, which are then used to forward these packets along a chosen explicit Label Switching Path (LSP). In this case, the conventional shortest path based routing infrastructure (e.g., OSPF) is overridden with an MPLS-based explicit routing overlay. While MPLS is a powerful technology for creating overlay networks to support any specific routing strategy, it is also expensive and suffers potentially from scalability problem in terms of LSP state maintenance. On the other hand, the advent of pure IP-based TE solutions challenges MPLS-based approaches in that Internet traffic can also be effectively tuned through native hop-by-hop routing, without the associated complexity and cost of MPLS. Some research works have indicated that OSPF/IS-IS link weights can be intelligently pre-assigned to achieve near-optimal path selections with respect to the expected traffic demand [4], [9].

Despite the progress for unicast services, traffic engineering for multicast flows remains largely a dark area. In the past few years, MPLS-based multicast TE has become a subject of interest, with a number of relevant research works becoming

available [2], [5], [6]. In contrast, pure IP-based multicast traffic engineering without MPLS overlay has not yet been explored. The reason for this situation can be summarized as follows. First, the Protocol Independent Multicast – Sparse Mode (PIM-SM) [3] uses the underlying IP unicast routing table for the construction of multicast trees, and hence it is difficult to decouple multicast traffic engineering from its unicast counterpart. Bandwidth optimization for multicast traffic can be formulated as the directed Steiner tree problem, which is *NP*-complete. The enforcement of Steiner trees can be achieved through packet encapsulation and explicit routing mechanisms such as MPLS tunneling. However, this approach lacks support from hop-by-hop protocols, due to Reverse Path Forwarding (RPF) in the IP multicast routing protocol family. In PIM-SM, if multicast packets are not received on the shortest path through which unicast traffic is delivered back to the source, they are discarded for avoiding traffic loops.

In this paper we investigate the feasibility of engineering multicast traffic based on plain IP routing protocols. Our objective is to minimize the overall network resource consumption with bandwidth constraints, in order to accommodate as many multicast sessions as possible. The enforcement of engineered PIM-SM path selections is via setting optimized link weights for the underlying link state routing protocols. In our proposed approach, PIM-SM follows the shortest path according to the pre-set link weights, whereas the resulting multicast tree is in effect a hop-count Steiner tree with minimum number of links involved, which implies that minimum bandwidth resources are consumed. We demonstrate this with the simple example of Figure 1. We assume that node *A* is the root of group *X* that contains member nodes *E*, *F* and *G*. If PIM-SM performs hop-count based shortest path (SP) routing, the total bandwidth consumed is 6 units (1 unit for each on-tree link), as shown in Figure 1(a). In effect, by applying Steiner tree heuristics to this simple example, it is easy to obtain the optimized multicast tree with 4 units of bandwidth consumption, as shown in Figure 1(b). This hop-count Steiner tree can be supported using explicit routing approaches such as MPLS tunnels. For example, in order to deliver multicast packets from node *A* to *E* via the engineered path, an LSP tunnel has to be set up along the non-shortest path $A \rightarrow C \rightarrow F \rightarrow E$. On the other hand, by intelligently assigning link weights for the underlying link-state IGP protocol, we can still achieve the same effect in terms of bandwidth conservation, as PIM-SM join requests follow the shortest path in terms of this set of link weights (Figure 1(c)). From this example we can see that hop-count Steiner tree based multicast traffic engineering can be reduced to plain shortest path routing by introducing a set of optimized link weights. The advantage is that, through link weight setting as calculated by off-line network provisioning, IP routers are able to construct optimized multicast trees by simply using Dijkstra's shortest path algorithm. Currently, one difficulty in implementation of this scheme is that most unicast routing protocols such as OSPF and IS-IS do not provide independent set of link weights for different types of flows. Hence it is undesirable to set link weights exclusively for multicast traffic engineering, without considering unicast traffic in the network. In order to decouple multicast from unicast path selection, our approach is based on the Multi-topology extension of the IS-IS protocol (M-ISIS) [7], which is able to populate dedicated Multicast Routing Information Bases (MRIBs, i.e. RPF tables) for PIM-SM routing. This multi-topology routing feature provides a

mechanism to separate TE for multicast and unicast flows. A detailed description of the M-ISIS based multicast TE framework will be presented in section 3.



(a) Hop-count based SP routing

(b) Steiner tree based Explicit routing

(c) Link weight based SP routing

Fig. 1. Multicast routing using different approaches

The optimization of link weights through shortest path routing for indirectly obtaining one single Steiner tree in terms of hop-counts is *NP-complete*, since this is an adapted version of the classical Steiner tree problem. In effect, a more practical problem concerning an Internet Service Provider (ISP) for multicast traffic engineering is how to assign a set of unified link weights, such that *all* the multicast trees within the network consume minimum bandwidth resources. At the same time, we consider an additional constraint that the total bandwidth allocated on each link for the overlapping multicast trees should not exceed its capacity. In this paper we adopt a Genetic Algorithm (*GA*) approach as off-line multicast traffic engineering for optimizing overall bandwidth consumption for multiple multicast flows. More specifically, the M-ISIS link weights are adjusted in each *GA* generation so that the overall fitness is geared towards optimized network resource consumption with the constraint of link capacity. The key novelty of this work is that, in a similar fashion to the work in [4], [9] for unicast traffic, multicast flows can also be optimized in hop-by-hop routing based IP networks without relying on MPLS tunneling.

2 Related Work

In [9], the authors proved that, any arbitrary set of *loop-free* routes can be represented with shortest paths with respect to a set of positive link weights. As a typical application to this conclusion, the authors of [4] claimed that by optimizing OSPF/IS-IS link weights for the purpose of load balancing, link congestion can be effectively avoided for unicast services. The key idea of the proposed algorithm is to intelligently adjust the weight of a certain number of links that depart from one particular node, so that new paths with equal cost are created from this node towards the destination. As a result, the traffic originally traveling through one single path can be split into other paths with equal OSPF/IS-IS weights.

Recently, research efforts have also addressed traffic engineering multicast flows, particularly for Quality of Service (QoS) and bandwidth optimization purposes. One common aspect of those schemes is that they are based on explicit routing, typically

through MPLS tunneling. The problem of bandwidth optimization in multicast routing is formulated as the Steiner tree problem, which has been extensively studied in the literature, with the TM heuristic [8] being a near-optimal solution. As already mentioned, Steiner trees can be enforced through point-to-multipoint LSPs. In [2], Steiner tree based heuristics are applied for computing multicast path selection only at the edge of MPLS domains, so that multicast TE within the network can be reduced to a unicast problem. In [5], the authors propose an online multicast TE scheme using Steiner tree heuristics, while also attempting to minimize multicast flow interferences. Despite its flexibility, the explicit routing overlay approach suffers from complexity and cost associated with MPLS deployment. This problem becomes more serious in multicast services, since point-to-multipoint LSPs need to be maintained. Taking these facts into account, we propose a novel scheme for engineering multicast flows in the pure IP based environment, without the deployment of MPLS tunneling.

3 Proposed M-ISIS Based Multicast TE

The traditional OSPF and IS-IS protocols only have uni-dimensional viewpoint on the weight of each link in the network, and this influences path selections for both unicast and multicast traffic. In contrast, M-ISIS provides the original IS-IS protocol with the additional ability of viewing the weight of each link for different logical IP topologies independently. For IPv4 multicast traffic, the field of Multi Topology identifier (MT-ID) with value 3 in M-ISIS is dedicated to the multicast reverse path forwarding topology, i.e., the RPF table for PIM-SM can be populated using a set of independent link weights with MT-ID equal to 3. With this multi-topology capability, it becomes possible that PIM-SM based multicast routing is completely decoupled from the underlying routing table for unicast traffic.

Figure 2 illustrates the basic framework of IP multicast traffic engineering through optimized M-ISIS link weight setting. First, the network topology (e.g., link capacity, edge router information) and the multicast "traffic matrix" are obtained as the input parameters for calculating the optimized link weights over an existing physical network infrastructure. The multicast traffic matrix can be derived through obtaining the following information from each group session: (1) bandwidth demand, (2) root node (i.e., ingress router) and a set of egress routers with potential receivers. An ISP can obtain this information from Service Level Agreements (SLAs) with customers. Here we assume the following business relationship: content providers have SLAs with an ISP and receivers subscribe multicast services offered by the content provider. The latter may pass the necessary information mentioned above to the ISP in order to aid multicast traffic matrix generation. In general, accurate multicast traffic matrix generation is a new research issue that needs further study, and it is outside the scope of this paper. Based on the multicast traffic matrix, the optimized link weights are computed through off-line algorithms (see sections 4 and 5) and configured in the routers that run the M-ISIS routing protocol with MT-ID equal to 3, which is dedicated to the multicast RPF table construction. On receiving Link State Advertisements (LSAs), each M-ISIS aware router computes shortest path trees according to this set of link weights and decides the NEXT_HOP router for a specific IP address/prefix. When a PIM-SM join request is received, the router simply looks

up the RPF table and finds the proper NEXT_HOP for forwarding the packet. In this scenario, the delivery of PIM-SM group join requests follows an engineered path, thus the resulting multicast distribution tree from the root to individual members conforms to the TE requirement. In addition, the multicast Forwarding Information Base (FIB) is dynamically updated for the incoming interface (iif) and outgoing interface (oif) list of each group. We can see from Figure 2 that, apart from the offline calculation and setting of link weights, there is no need for any other configuration or extensions to the current M-ISIS and PIM-SM protocols for multicast traffic engineering purposes.

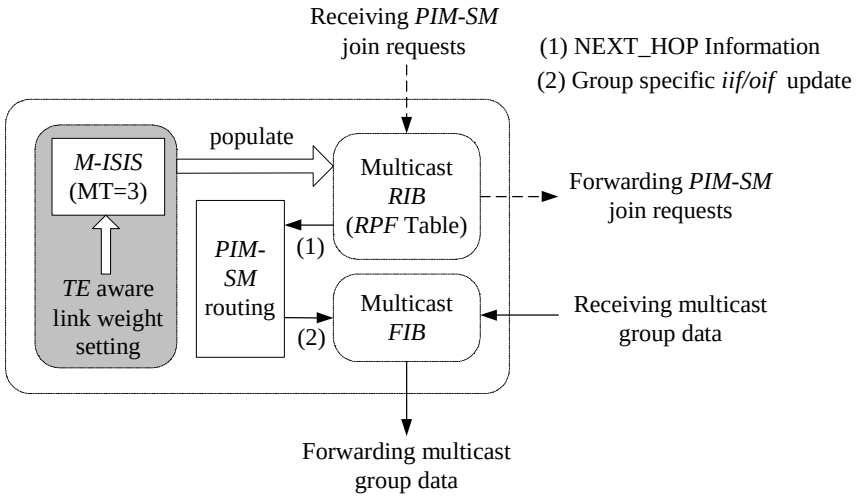


Fig. 2. *M-ISIS* based multicast traffic engineering

4 Problem Formulation

The following is the integer-programming formulation for computing bandwidth constrained Steiner trees in terms hop counts with the objective of minimizing overall bandwidth consumption. By setting the group-specific *binary* variables $x_{ij}^{g,k}$ and y_{ij}^{g} for each link (*i, j*), a set of explicit multicast trees with minimum number of links is obtained, which implies that minimum bandwidth consumption is achieved. We first present some definitions below:

G — Total number of active multicast groups;

r_g — Root node of group g ($g = 1, \ldots, G$);

V_g — Multicast member (receiver) set for group g;

D_g — Bandwidth demand for group g traffic on each link;

C_{ij} — Bandwidth capacity of link (i, j);

y_{ij}^{g} — Equal to 1 if link (i, j) is included in the multicast tree for group g;

$x_{ij}^{g,k}$ — Equal to 1 if link (i, j) is on the unique elementary path from the root node r_g of group g to the group member node k in the multicast tree.

The integer-programming problem of computing a set of bandwidth constrained Steiner trees with minimum overall bandwidth consumption is formulated as:

Minimize

$$\sum_{g=1}^{G} \sum_{(i,j)\in E} D_g \times y_{ij}^{g}$$

Subject to

$$\sum_{h\in V} x_{ih}^{g,k} - \sum_{j\in V} x_{ji}^{g,k} = \begin{cases} 1 & i = r_g \\ -1 & i = k, \quad k \in V_g \\ 0 & i \neq r_g, i \notin V_g \end{cases} \tag{1}$$

$$x_{ij}^{g,k} \leq y_{ij}^{g} \qquad (i,j) \in E,\ k \in V_g \tag{2}$$

$$x_{ij}^{g,k} = 0,1 \qquad (i,j) \in E,\ k \in V_g \tag{3}$$

$$y_{ij}^{g} = 0,1 \qquad (i,j) \in E \tag{4}$$

$$\sum_{g=1}^{G} y_{ij}^{g} \times D_g \leq C_{ij} \qquad (i,j) \in E \tag{5}$$

The variables to be determined are $x_{ij}^{g,k}$ and y_{ij}^{g} for every link $(i, j) \in E$. Constraint (1) ensures one unit of multicast flow from r_g to every group member node $K \in V_g$. Constraint (2) guarantees that the amount of flows along link (i, j) must be zero if this link is not included in the multicast tree for group g. Variables $x_{ij}^{g,k}$ and y_{ij}^{g} are confined to binary values in constraints (3) and (4) for non-splitting of multicast flows. Finally it is required in (5) that the total bandwidth consumption on each link should not exceed its capacity.

As we have mentioned before, the enforcement of the above set of bandwidth constrained hop-count Steiner trees can be achieved through an explicit routing overlay, e.g. through MPLS tunneling, on per group basis. However, the paths in the Steiner tree from r_g to individual group members $K \in V_g$ might not completely overlap with the shortest paths between them. This means that, in case of hop-by-hop routing, multicast traffic flowing on the Steiner tree will be discarded due to the RPF check failure, if the packets are not received from the correct interface on the shortest path back to the source. In order to apply the above programming model to IP layer solutions, we introduce a unified M-ISIS link weight w for each link (i, j), and by properly setting those link weights it is guaranteed that the tree branch from r_g to each receiver $K \in V_g$ is the shortest path according to this set of weights. Put in other words, our strategy is to represent this set of explicit hop-count Steiner trees with shortest path trees through intelligent configuration of a unified set of link weights.

5 A Genetic Algorithm (GA) Based Solution

5.1 Encoding and Initial Population

In our *GA* approach each chromosome is represented by a link weight vector $W = <w_1, \ldots w_{|E|}>$ where $|E|$ is the total number of links in the network. The value of each weight is within the range from 1 to *MAX_WEIGHT*. In our experiments we define the value of *MAX_WEIGHT* to be 64 for reducing the search space. On the other hand, the population size is set to 100, with the initial values inside each chromosome randomly varying from 1 to *MAX_WEIGHT*. In addition to these randomly generated chromosomes, we add the solution of using hop-count as the link weight into the initial population (i.e., the weight of every link is set to 1). This is to guarantee that every link can potentially obtain the lowest link weight such that it has the chance to be included into the resulting trees.

5.2 Fitness Evaluation

Chromosomes are selected according to their *fitness*. In our approach, the bandwidth constraint is embedded into the fitness function as a penalty factor, such that the search space is explored with the potential feasible solutions. The fitness of each chromosome can be defined to be a two-dimensional function of the overall network load ($l1$) and excessive bandwidth allocated to overloaded links ($l2$), i.e.,

$$fitness = f(l1, l2) = \frac{\mu}{\alpha \times l1 + \beta \times l2} \tag{6}$$

where α, β and μ are manually configured coefficients.
In equation (6) $l1$ and $l2$ are expressed as follows:

$$l1 = \sum_{g=1}^{G} \sum_{(i,j) \in E} D_g \times y_{ij}^g \tag{7}$$

$$l2 = \sum_{(i,j) \in E} \omega_{ij} \times (\sum_{g=1}^{G} D_g \times y_{ij}^g - C_{ij}) \tag{8}$$

where

$$\omega_{ij} = \begin{cases} 0 & if \quad \sum_{g=1}^{G} D_g \times y_{ij}^g \le C_{ij} \\ 1 & otherwise \end{cases} \tag{9}$$

We note from fitness function (6) that the objective is two fold: first, chromosomes of the new generations should converge towards a set of Steiner trees in terms of hop counts with the lowest bandwidth consumption, and second, solutions obtained from the offspring should be feasible in that the total bandwidth allocated to the multicast flows traveling through each link should not exceed its capacity. The tuning of α and β can be regarded as a tradeoff between overall bandwidth conservation and load balancing. For example, if we let $\beta = 0$ then the objective is to conserve bandwidth

resources only, while setting $\alpha = 0$ infers to minimize link overloading within the network.

```
Procedure Computing_fitness(Chromosome i)
begin
      Set the weight of each link according to the gene
        values in chromosome i;
      for each multicast group g
            Compute the shortest path tree T_g rooted at
              r_g, and spanning to all members in V_g;
            for each link (u, v) in T_g
                  Update link load L_uv according to the
                    bandwidth demand D_g of group g;
      end for
      Load1 = 0;  Load2 = 0;
      for each link (u, v) in the network
            Load1 = Load1 + L_uv;
            If L_uv > C_uv
                  Load2 = Load2 + (L_uv - C_uv);
      end for
      return fitness = f(Load1, Load2);
end
```

Fig. 3. Fitness calculation

5.3 Crossover and Mutation

According to the basic principle of Genetic Algorithms, chromosomes with better fitness value have higher probability of being inherited into the next generation. To achieve this, we first rank all the chromosomes in descending order according to their fitness, i.e., the chromosomes with high fitness (lower overall load) are placed on the top of the ranking list. Thereafter, we partition this list into two disjoined sets, with the top 50 chromosomes belonging to the upper class (*UC*) and the bottom 50 chromosomes to the lower class (*LC*). During the crossover procedure, we select one parent chromosome C_U^i from *UC* and the other parent C_L^i from *LC* in generation i for creating the child C^{i+1} in generation i+1. Specifically, we use a crossover probability threshold $K_C \in (0, 0.5)$ to decide the genes of which parent to be inherited into the child chromosome in the next generation. We also introduce a mutation probability threshold K_M to randomly replace some old genes with new ones. In addition to this type of conventional mutation, we also find the *congested* link with the highest load in the chromosome of the new generation, and we randomly raise its link weight in an ad hoc manner so as to avoid hot spots. In non-congested conditions, this type of mutation the highest loaded link is suppressed.

```
Procedure Crossover(C_U^i , C_L^i)
begin
      for all genes j = 1 … |E|
            Generate r = random(MAX_WEIGHT);
            if r > K_C
                  C^{i+1}(j) = C_U^i(j);
            else if r > K_M
                  C^{i+1}(j) = C_L^i(j);
            else
                  C^{i+1} = random[1, MAX_WEIGHT];
      end For
      Find gene (link) t with the highest load in C^{i+1};
      if  L_t (link load)  > Cap_t (link capacity)
            C^{i+1}(t) = random[C^{i+1}(t), MAX_WEIGHT];
      return C^{i+1};
end
```

Fig. 4. Crossover and mutation

6 Simulation Results

6.1 Simulation Configuration

In our simulation, we adopt the Waxman's model in GT-ITM topology generator for constructing our network. We generate a random graph of 100 nodes, out of which 50 are configured as Designated Routers (DRs) with attached group sources or receivers. The scaled bandwidth capacity of each network link is set to 10^5 units.

The simulation parameters of the proposed Genetic Algorithm are illustrated in table 1. Apart from the GA approach, we also implemented two non-TE based hop-by-hop routing paradigms and one explicit routing approach: (1) shortest path routing with random link weight setting (Random), (2) shortest path routing in terms of hop-counts (SPH), and (3) Steiner tree approach using the TM heuristic [8]. For this TM Steiner tree algorithm, we use hop count as the link weight, and the resulting trees are group specific, i.e., one Steiner tree is specifically constructed for each multicast group. In the next section we will show that the TM heuristic has the best performance among the four algorithms in terms of bandwidth conservation. Nevertheless, it should be emphasized that, this solution requires the setting up of MPLS tunnels for explicit routing on a per-group basis, and this cannot be directly achieved in a pure IP environment. Hence, the inclusion of the TM algorithm is only to use its performance as a lower bound reference for comparison with the other three hop-by-hop oriented approaches.

Table 1. GA parameter configuration

Parameter	Value	Parameter	Value
Population size (P)	100	μ	10^7
Maximum generation (*M*)	500	α	1.0
Maximum link weight (*MAX_WEIGHT*)	64	β	10
Crossover threshold (K_C)	0.30	Mutation threshold (K_M)	0. 01

6.2 Performance Evaluation

We found from our simulation that shortest path routing with hop-counts (SPH) has higher capability in finding *feasible solutions* (i.e., no overloaded links incurred) than random link weight setting approaches (shown later). Hence, we will start from the comparison between GA and SPH in the capability of exploring feasible solutions. Figure 5 presents the ratio of successful instances obtained by GA but failed to be found in SPH. We define the Maximum Link Overload Rate (*MLOR*) as follows:

$$MLOR = \max_{(i,j)\in E} \left(\frac{\sum_{g=1}^{G} D_g \times y_{ij}^g - C_{ij}}{C_{ij}} \right)$$

From this definition we can see that *MLOR* reflects the overloading scale of the most congested link (if any, i.e., *MLOR*>0). In the figure, when the value of *MLOR* computed by SPH is in range of (0%, 5%], GA can obtain feasible solutions (i.e. $MLOR_{GA} \leq 0$) for 65% of these instances. We can also see that, with the increase of external bandwidth demands, the capability of GA in finding feasible solutions is decreasing. When the *MLOR* value of SPH grows up to 25% due to the higher external traffic demand, the success rate of GA drops to 5%. From this figure, it can be inferred that, when the external group traffic demand is at the brink of causing network congestion, GA has higher capability of avoiding link overloading compared to other approaches. Obviously, it may be the case that no feasible solution exists at all, if external traffic demand exceeds a certain threshold.

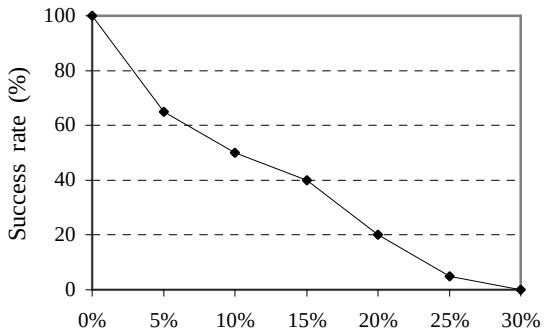


Fig. 5. GA success rate vs. $MLOR_{SPH}$

Figure 6 illustrates the feature of overall bandwidth conservation capability of individual schemes with the variation of maximum group traffic demand D_g. We see that the GA approach exhibits the best capability in conserving bandwidth among all the hop-by-hop routing schemes. Typically, when the network is under-utilized, our proposed GA approach exhibits significantly higher performance than the conventional IP based solutions without explicit routing. For example when D_g > 3000, the overall bandwidth consumption of the Random and SPH solutions are higher than that of GA by 19.3% and 14.9% respectively. Compared with the TM heuristic that needs support from MPLS overlaying, the gap from GA is below 8%. However, when the external traffic demand grows, the performance of GA converges to that of the SPH approach. On the other hand, although the TM algorithm exhibits significant higher capability in bandwidth conservation when the external traffic demand grows (D_g > 4000) this does not mean what have been obtained are feasible solutions without introducing overloaded links.

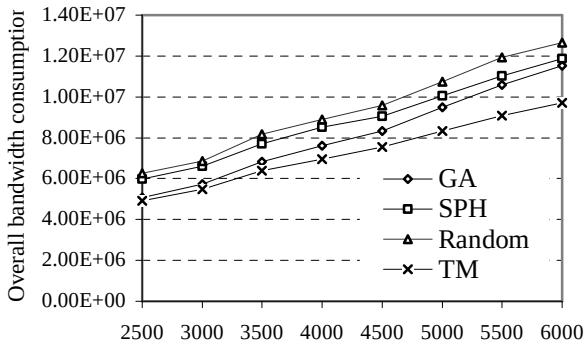
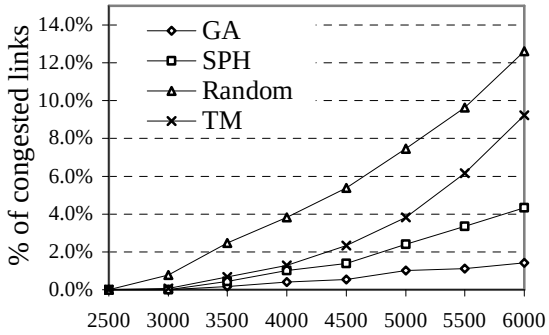


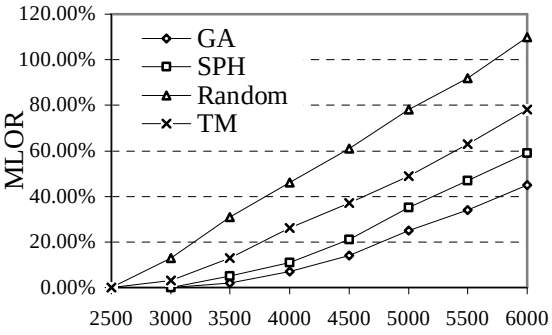
Fig. 6. Total bandwidth consumption vs. Max D_g

The rest of the simulation evaluates the capability of alleviating network congestions in our proposed solution. Figure 7(a) shows the relationship between the proportion of overloaded links and the maximum group traffic demand D_g in time of congestion. From the figure we can see that there are more overloaded links as D_g increases. The most interesting result is that, through our GA optimization, the percentage of overloaded links is significantly lower than all the other routing schemes. In the most congested situation (D_g =6000), the average rate of overloaded links computed by GA is only 1.4%, in contrast to 12.6% by random link weight setting, 8.6% by the TM heuristic, and 4.4% by SPH respectively. On the other hand, the amount of overloaded bandwidth occurred on the most congested links is another important parameter an ISP is interested in. An ISP should avoid configuring the network resulting in hot spots with high *MLOR*. Through our simulations, we also find that the proposed GA approach achieves the lowest *MLOR* performance. In Figure 7(b), the overloading scale is 45% of the bandwidth capacity on the most congested link in the GA approach with D_g equal to 6000, while this value reaches 110% and 59% in random

link weight setting and SPH respectively. Even by using explicit routing TM heuristic, the overloaded bandwidth is 78% of the original link capacity.



(a) Overloaded link rate vs. Max D_g



(b) *MLOR* vs. Max D_g

Fig. 7. Link congestion comparisons

7 Summary

In this paper we proposed an efficient scheme for offline IP layer multicast traffic engineering with bandwidth constraints using Genetic Algorithms. By means of off-line optimizing and pre-configuring M-ISIS link weights, traditional Steiner tree based multicast traffic engineering can be reduced to plain PIM-SM shortest path routing that is widely supported in the current IP routers. Moreover, the GA based approach also exhibits higher capability in finding feasible solutions and reducing network congestion. As far as we know, our proposed approach represents the first attempt to explore effective solutions to multicast traffic engineering based on the hop-by-hop routing semantics. This is in contrast to most of the current multicast traffic engineering schemes that require MPLS support.

Our future work will address the problem of M-ISIS link weight optimization in case of significant traffic dynamics and topology changes (e.g., link failure). Relevant research has been carried out for unicast flows, and we believe that it is equally important to consider this relatively dynamic scenario in IP based multicast traffic engineering semantics.

References

1. D. Awduche et al, "Overview and Principles of Internet Traffic Engineering", RFC 3272, (2002)
2. Baijian Yang et al, "Multicasting in MPLS Domains", Computer Communications, Vol. 27(2), 162-170
3. B. Fenner, "Protocol Independent Multicast - Sparse Mode (PIM-SM): Protocol Specification (Revised)", draft-ietf-pim-sm-v2-new-09.txt, (2004), work in progress
4. B. Fortz et al, "Internet Traffic Engineering by Optimizing OSPF Weights", Proc. IEEE INFOCOM (2000)
5. M. Kodialam et al, "Online Multicast Routing with Bandwidth Guarantees: A new Approach Using Multicast Network Flow", IEEE/ACM Trans. on Networking, Vol. 11, No. 4, (2003), 676-686
6. D. Ooms et al, "Overview of IP Multicast in a Multi-Protocol Label Switching (MPLS) Environment", RFC 3353 (2002)
7. T. Przygienda et al, "M-ISIS: Multi Topology (MT) Routing in IS-IS", draft-ietf-isis-wg-multi-topology-07.txt, (2004), work in progress
8. H. Takahashi, A. Matsuyama, "An Approximate Solution for the Steiner Problem in Graphs", Math. Japonica 6, 533-577
9. Y. Wang et al, "Internet Traffic Engineering without Full Mesh Overlaying", Proc. IEEE INFOCOM (2001)