Work in Progress: Bandwidth Optimization for Multicast Transmissions in Virtual Circuit Networks

Vincent Reinhard[1], Joanna Tomasik[1], Dominique Barth[2], and Marc-Antoine Weisser[1]

[1] Computer Science Department, SUPELEC, 91192 Gif sur Yvette, France
{vincent.reinhard,joanna.tomasik,marc-antoine.weisser}@supelec.fr
[2] PRiSM, Université de Versailles-St-Quentin. Versailles, France.
dominique.barth@prism.uvsq.fr

Abstract. The CARRIOCAS project aims to guarantee QoS connectivity services to distributed applications in a Telecom carrier network. A large number of these applications (for example video applications) use a multicast service packet delivery. Multicast which minimizes the total used bandwidth in the MPLS network has become an important subject. We study multicast routing in the network where only some routers can duplicate packets. We prove that the construction of a multicast tree minimizing the bandwidth used in such a network is a *NP*-complete problem and we propose an heuristic algorithm to solve it. We evaluate the performance of the heuristic in terms of total bandwidth used by the multicast for different network sizes.

Keywords: Multicast, bandwidth optimization, MPLS networks

1 Introduction

As distributed applications become more and more popular, Internet providers have become interested in furnishing to their customers not only the connectivity service but other services (storage, computation) as well. Supplying at the same time connectivity and other services was one of the motivations of the CARRIOCAS project [1][2][3][4] The latter is one of the projects of the "SYSTEM@TIC PARIS-REGION [3]competitiveness cluster". It aims to provide connectivity services and usual grid services to enable the executions of distributed application workflows in a Telecom carrier network. In the CARRIOCAS service architecture, the network service management is centralized per a network operator domain in order to enable querying for explicit bandwidth reservation. This network service management disposes of the network topology. It computes a virtual connection for each external connectivity service demand and, whenever possible, reserves a bandwidth on the connection. If no virtual connection

[3] SYSTEM@TIC PARIS-REGION and the CARRIOCAS project are supported by the French Ministry of Industry, Essonnes, Haut-de-Seines and Paris General Council.

with the requested bandwidth can be reserved, it rejects the reservation query. This mechanism ensures that the bandwidth needed for the CARRIOCAS applications never exceeds the bandwidth available and guarantee the QoS for applications. As GMPLS [14] is to be deployed in the CARRIOCAS network, it has to ensure both unicast and multicast communications.

There are several schemes for multicasting data in networks [6][7]. A first one is to construct virtual circuits from the multicast source to each destination. Such scheme is equivalent to multiple unicasts and the network bandwidth used by a large multicast group may become unacceptable [8]. In another scheme the multicast source sends data to the first destination and each destination acts as a source for the next destination until all destinations receive the data flow. In yet another scheme intermediate routers make copies of data packets and dispatch them to their successors in the multicast tree. This solution allows the multicast transmission to share bandwidth on the common links. Many multicast tree algorithms have been proposed and can roughly be classified into two categories [6]. The first category contains the algorithms based on the shortest path while minimizing the cost of the path from the multicast source to each destination. The second category contains algorithms based on the Steiner tree problem [9][10][11][12]. Such algorithms derived from the Steiner tree problem minimize the total cost of the multicast tree. This minimization is a *NP*-complete problem [10].

In this paper we study a solution based on the last mentioned scheme used in a MPLS [15] optical network which introduces an additional strong constraint on the construction of a multicast tree. From the technological point of view, routers able to duplicate packets introduce a supplementary delay due to O/E/O conversions and are more expensive. For these reasons network operators want to limit the number of such routers which we call "diffusing nodes". The constraint on the number of diffusing nodes invalidates the existing algorithms of multicast tree construction based on the Steiner tree [9][10] because they were proposed for networks the routers of which can all duplicate data. Such a constraint has already been considered for wavelength-routed networks [5] under the condition that all routers support the "drop and continue" functionality. Moreover, no complexity study had been made. In our work we consider that the routers do not support "drop and continue". We prove that minimizing the total cost of the multicast tree under this condition and with the constraint on the number of diffusing nodes is an *NP*-complete problem and propose an heuristic algorithm to solve it.

The rest of this paper is organized as follows. In Section II and III we propose a network model and define the bandwidth optimization problem. In Section IV, we present a bipartite graph used to study the bandwidth optimization problem. In Section V we prove that the problem is *NP*-complete and Section VI contains a pseudo-polynomial algorithm to solve it. We also propose an heuristic algorithm to solve the bandwidth optimization problem in Section VII. Section VIII contains our results concerning the performances of the heuristic. Finally, we conclude and outline future work.

2 Network Modeling

We model a network using a directed symmetric graph $G = (V, E)$. We define $cap(u, v)$ as the bandwidth available on the link (u, v), $u, v \in V$. A multicast request is a triplet $\epsilon = (e, R, c)$, with $e \in V$ the source of the multicast, $R \subset V$ the set of destinations and c the size of the data flow. In this work, we consider only multicast requests of the size $c = 1$ and note them $\epsilon = (e, R)$. We define $D_G \subset V$ as the set of diffusing nodes of G. An example of a network graph with a multicast request is given in Fig. 1.

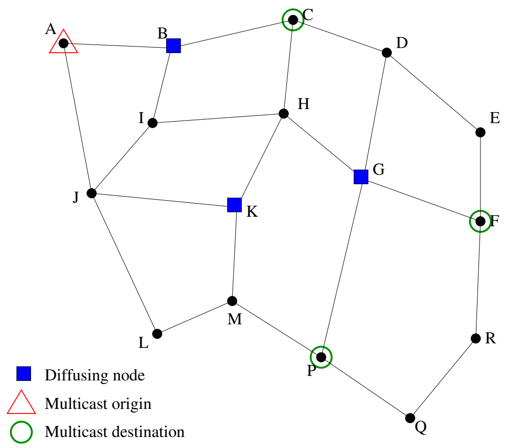

Fig. 1. A network graph G, $D_G = \{B, K, G\}$ and a request $\epsilon = (A, \{C, F, P\})$.

3 Definition of the Bandwidth Optimization Problem

Our goal is to optimize the bandwidth used by multicasts in the network represented by graph G. We define a diffusion tree for $\epsilon = (e, R)$ as an arborescence A_ϵ in G rooted in e, spanning R and with all leaves included in R (Fig. 2). We define $V(A_\epsilon)$ as the set of nodes of A_ϵ and $E(A_\epsilon)$ as the set of edges of A_ϵ. A request ϵ is satisfied by the set of paths C_{A_ϵ} in A_ϵ, defined as follows:

- every node of R is the final extremity of exactly one path in C_{A_ϵ},
- every node of D_G is the final extremity of at the most one path in C_{A_ϵ},
- the origin of a path in C_{A_ϵ} is either e or a node of D_G, In the latter case, the node of D_G is also the final extremity of a path in C_{A_ϵ}.
- a node of D_G is in a path $p \in C_{A_\epsilon}$ only if it is the final extremity or the origin of p.

The load $ch_{A_\epsilon}(u, v)$ of a link $(u, v) \in E(A_\epsilon)$ is the number of paths $p \in C_{A_\epsilon}$ such as (u, v) is a link of p. The load ch_{A_ϵ} of the arborescence A_ϵ is $\sum_{(u,v)\in E(A_\epsilon)} ch_{A_\epsilon}(u, v)$.

Definition of the problem *diff_ tree*:
Given a network G, its set of diffusing nodes D_G, and a multicast request $\epsilon = (e, R)$, find an arborescence A_ϵ for ϵ, such as ch_{A_ϵ} is minimal.

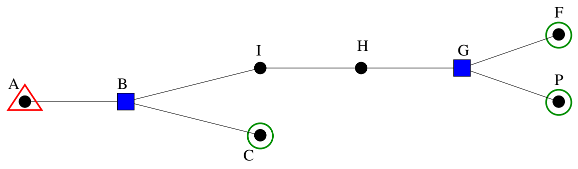

Fig. 2. A possible arborescence A_ϵ for G and ϵ shown in Fig. 1.

4 Complexity of the Problem

We study the complexity of *diff_ tree* and we prove that its decision problem *diff_ tree_ dec* is *NP*-complete. First, we prove that the certificate of the decision problem is in class P. We then reduce the problem of the weighted set cover which is *NP*-complete [17] to the problem *diff_ tree_ dec*.

First, we show that the certificate of the problem *diff_ tree_ dec* is in class P. Given an instance of the problem *diff_ tree_ dec* (a graph of the network, a set of diffusing nodes and a multicast request) and an associated arborescence A_ϵ, we can compute ch_{A_ϵ} in a polynomial time (we must compute a finite set of shortest paths). The certificate of the problem *diff_ tree_ dec* is therefore in class P. We now reduce the problem of the weighted set cover which is *NP*-complete [17] to the problem *diff_ tree_ dec* and we show that the solution to the problem of the set cover is equivalent to the solution to *diff_ tree_ dec*.

Definition of the weighted set cover problem:
Given a collection C of subsets of a finite set S, with a cost h associated to each subset and an integer K, is it possible to find a cover S of size less than K, namely, a collection $C' \subseteq C$ such as every element of S is in at least one subset C' and whose cost is less than K.

Let I be an instance of the set cover problem, namely a collection $C = \{c_1, ..., c_m\}$ of subsets of a finite set $S = \{s_1, ..., s_n\}$, a cost h associated to each subset c_i and an integer K. From this instance I, we can construct in polynomial time an instance I' of the problem *diff_ tree_ dec* (Fig. 3): Let $G = (V, E)$ be a graph. We define V as the set of its nodes followingly: there is a node for each subset c_i from collection C; a node for each element s_i from finite set S and a node e. We define E as the set of its edges created as follows: an edge of weight $l > h/2$ between any node representing c_i from collection C and every node representing an element s_i of the c_i; an edge of weight h between the node e

and each node representing a subset c_i. We define $D_G = \{c_1, ..., c_m\}$ and K' an integer which adequate value will be defined later. Let $\epsilon = (e, R)$ be a multicast request in G, with $R = S$.

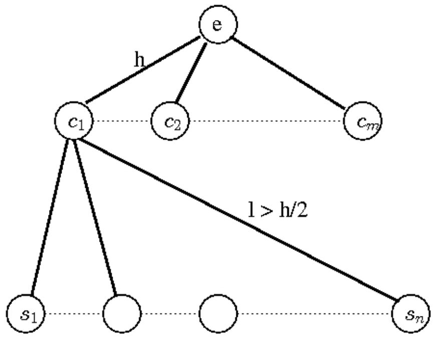

Fig. 3. Construction of an instance I' of the problem *diff_ tree_ dec* from an instance I of the problem of the set cover.

In a solution to I' in G, we know that a node of D_G is never reached from another node of D_G but always from node e. Since $l > h/2$, the weight of a path between two nodes of D_G is always greater than $2l$, whereas the weight of any arc between e and a node of D_G is $h < 2l$. A solution to I' in G is an arborescence A_ϵ rooted in e. A_ϵ contains all nodes s_i and $j \leq m$ nodes c_i such as $ch_{A_\epsilon} < K'$. Futhermore, $ch_{A_\epsilon} = j * h + l * n$ (the weight of the j edges between node e and nodes c_i and the weight of the n edges between nodes c_i and s_i). In instance I, selecting the subsets c_i corresponding to the nodes c_i which are also nodes of A_ϵ in I' gives a cover S of size $K = j * h = K' - l * n$.

Therefore, if there is a solution of weight less than K' to the instance I', then we can construct a solution of weight less than $K = K' - n * l$ to the instance I. Similarly, if there is a solution of weight less than K to the instance I, then we can construct a solution of weight less than $K' = K + n * l$ to the instance I'. We can reduce any instance of the problem of the set cover to an instance of the problem *diff_ tree_ dec* in a polynomial time. Furthermore, we have demonstrated that the certificate of this problem is in class P. Therefore, the problem *diff_ tree_ dec* is NP-complete.

5 The Diffusion Graph

To treat a multicast request ϵ in graph G, we construct a new directed and weighted graph containing the three sets of nodes: e, D_G and R. We call this graph a "diffusion graph" and define it as $B_{G,(e,R)}$. The weights of its arcs correspond to the weights of shortest paths in G. The graph $B_{G,(e,R)} = (V_B, E_B)$ is defined as follows:

- V_B the set of nodes of $B_{G,(e,R)}$ such as $V_B = \{e\} \cup R \cup D_G$.
- E_B the set of arcs constructed as follows:
  - $\forall u \in V_B$, $u \neq e$, the arc $(e, u) \in E_B$.
  - $\forall (u, v) \in D_G$, the arcs (u, v) and $(v, u) \in E_B$.
  - $\forall u \in D_G$, $\forall v \in R$, the arc $(u, v) \in E_B$.
- The weight of the arc (u, v) of E_B is the weight of a shortest path from u to v in $G \setminus \{D_G \setminus \{D_G \cap \{u, v\}\}\}$. If there is no path from u to v in $G \setminus \{D_G \setminus \{D_G \cap \{u, v\}\}\}$, the arc does not exist.

Fig. 4 represents the diffusion graph for the network graph from Fig. 1. The complexity of the diffusion graph construction is $O(n^3)$, where n is the number of vertices in G.

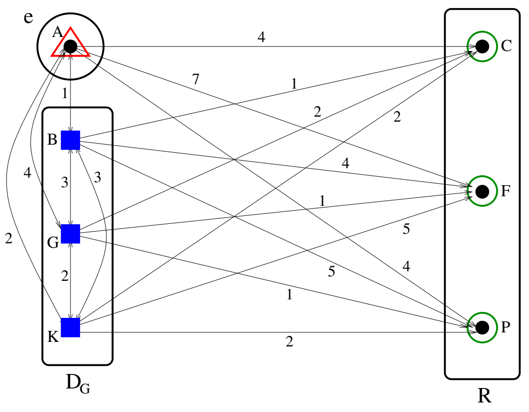

Fig. 4. The diffusing graph $B_{G,(e,R)}$ for G and the multicast request of Fig. 1.

We prove that finding a solution to the *diff_tree* problem in graph G is equivalent to finding a solution to the *diff_tree* in the diffusion graph.

To find a solution to the *diff_tree* problem in graph G, we search an arborescence A_ϵ such as ch_{A_ϵ} is minimal. In the diffusion graph $B_{G,(e,R)}$, we search an arborescence B_ϵ such as ch_{B_ϵ} is minimal. Any arc (u, v) of $B_{G,(e,R)}$ represents a shortest path from u to v in G. If we know an arborescence B_ϵ in $B_{G,(e,R)}$ for which ch_{B_ϵ} is minimal, we can determine a set of paths C_{B_ϵ} in B_ϵ. We can construct a set of paths C_{A_ϵ} in G such as $ch_{A_\epsilon} = ch_{B_\epsilon}$ using the corresponding shortest paths of G represented by the arcs of $C_{A_{B_\epsilon}}$. From the set of paths C_{A_ϵ}, we can construct the corresponding arborescence A_ϵ in G. Since ch_{B_ϵ} is minimal, ch_{A_ϵ} is minimal as well. Similarly, if we know A_ϵ in G such as ch_{A_ϵ} is minimal, we can create B_ϵ in $B_{G,(e,R)}$ such as ch_{B_ϵ} is minimal.

6 A Pseudo-polynomial Exact Algorithm

We propose an exact algorithm to determine an arborescence A_ϵ minimizing ch_{A_ϵ} in $B_{G,(e,R)}$. Let us define $S \subseteq D_G$ as a set of diffusing nodes which are also nodes of A_ϵ. Knowing S allows us to construct A_ϵ in polynomial time as follows. Let $\{e\} \cup R \cup S$ be the nodes of A_ϵ. We chose for each node in R an arc of minimum weight which has as its origin a node of S and we add this arc to A_ϵ. We then compute a spanning tree in the clique formed by e and the nodes of S and we add it to A_ϵ. Any part of this algorithm is polynomial. To determine which nodes are in S, we successively generate all subsets of D_G. We construct for each subset a minimal arborescence (as described above). Finally, we chose as solution to the problem the arborescence with the minimal ch_{A_ϵ}.

The complexity of this exact algorithm depends on the cardinality of the power set of D_G and it is $O(2^{\#D_G})$. Fig. 5 shows the solution computed by this algorithm for the graph and request shown in Fig. 1.

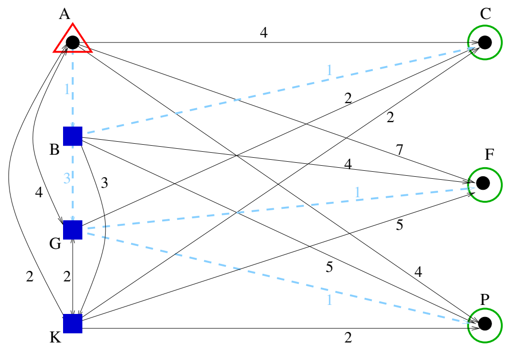

Fig. 5. Solution for the graph of Fig. 1 computed by the exact algorithm. $ch_{A_\epsilon} = 7$

7 Heuristic Algorithm

Since the pseudo-polynomial exact algorithm can not be used to solve the *diff_ tree* problem when the number of diffusing nodes is large, we propose a polynomial heuristic algorithm.

This heuristic is based on an algorithm of maximal flow of minimal cost and its description is given below. In the experiments that we performed, we used the Busacker and Gowen maximal flow of minimal cost algorithm (Busacker and Gowen, 1961). First, we construct a directed graph $F_{G,(e,R)} = (V_F, E_F)$ (Fig. 6) based on $B_{G,(e,R)}$ which will be used for a flow algorithm.

- V_F is the set of nodes of $F_{G,(e,R)}$ with $V_F = V_B \cup \{p\}$ where p is an additional node used as a sink.
- E_F is the set of arcs with capacities and costs constructed as follows:
  - $\forall (u, v) \in E_B$, $(u, v) \in E_F$, its cost is the weight of $(u, v) \in E_B$ and the capacity on (u, v) is infinite.
  - $\forall u \in D_G$, E_F contains (u, p). The cost of these arcs is always 0 and their capacity is 1.

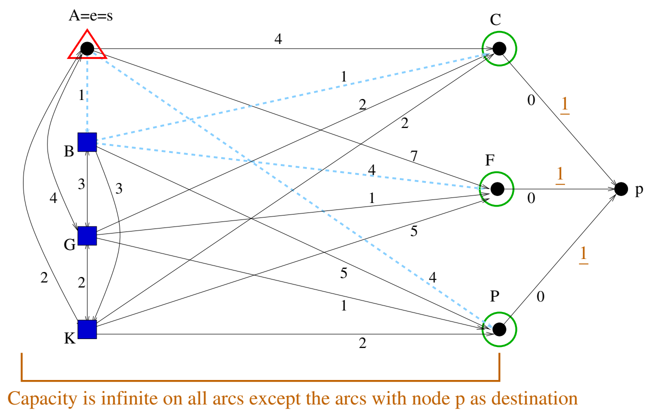
A=e=s
C
F
P
p
B
G
K
4
1
2
2
7
1
4
1
5
5
4
1
2
3
3
4
2
2
0
0
0
1
1
1
Capacity is infinite on all arcs except the arcs with node p as destination

Fig. 6. $F_{G,(e,R)}$ constructed from $B_{G,(e,R)}$ of Fig. 4. Capacities are highlighted. The solution constructed by the heuristic is shown in dotted lines and $ch_{A_\epsilon} = 10$.

In $F_{G,(e,R)}$ we compute a maximal flow of minimal cost from e to p. Since the capacity on each arc (u, p) is 1, the maximal flow value is equal to the number of nodes in R and since a maximal flow has to pass by all $r \in R$ every node of R is in a flow. The computation of a maximal flow of minimal cost gives an arborescence in $F_{G,(e,R)}$ which contains e and all nodes of R. This arborescence deprived of the sink p is a possible A_ϵ solving the *diff_ tree* problem in $B_{G,(e,R)}$. In any algorithm of maximal flow of minimal cost, the cost of each arc is counted as many times as the flow value on it. To compute ch_{A_ϵ} (Section III), we count the cost of each arc used in the solution only once. For this reason, we modify the maximal flow of minimal cost algorithm by setting to zero the cost of an arc which has already been used by a flow. Setting the cost of the arcs already used to zero impacts the computation of the solution. The arborescence A_ϵ corresponding to the maximal flow is not necessarily of minimal cost. A solution found by this heuristic for the graph of Fig. 1 is illustrated by Fig. 6.

The complexity of this heuristic is the same as the algorithm of maximal flow of minimal cost (for the Busacker and Gowen algorithm, $O(n^4)$).

8 Results

We performed experiments to study the performance of our heuristic in terms of the number of links used by the multicast tree which in our case corresponds to the bandwidth used by the multicast. We also studied the influence of the number of diffusing nodes on the performances of our algorithm.

We generate graphs $G = (V, E)$ representing a network with the BRITE generator [13] using the Waxman model (whith parameters: $\alpha = 0.15$, $\beta = 0.2$, $m = 2$, $MaxBW = 1024$, $MinBW = 10$). We define $T \subset V$ as the set of nodes of degree smaller than three.

In all experiments, to determine the location of the diffusing nodes in G we use a k-center heuristic algorithm based on a dominating set algorithm [16]. This approach was chosen because the k-center algorithm distributes the centers "equally" in the graph. In our opinion this approach is adapted to deal with multicasts whose origins are not a *priori* known efficiently.

We used the Busacker and Gowen algorithm as basis for our heuristic algorithm. The results obtained could be slightly different with another maximal flow minimal cost algorithm.

We generate multicast requests (multicast groups) in the network as follows. We chose an origin for multicasts uniformly in V. For each chosen origin, we generate different destination sets. For each destination set, we use a normal distribution $N_1 = \mathcal{N}(10\%\#T, 2\%\#T)$ to fix the number of destinations. The destinations are then chosen uniformly in T.

We started with a topology of 200 nodes. We arbitrarily fixed $\#D_G = 6$ and placed the diffusing nodes in the network according to the k-center algorithm. We chosed multicast sources and for each source generated 30 destination sets. We computed the average number of links used in the multicast trees constructed by both the exact algorithm and our heuristic. We constructed an histogram containing the results as follows. We created weight intervals and placed every generated multicast request in such an interval depending on the weight of its best multicast tree (constructed by the exact algorithm). We arbitrarily set the weight intervals length to 5. For each interval we computed the average weight of the multicast tree constructed by the exact algorithm and by the heuristic. We then showed in the histogram the relative average increase of the weight of the solution computed by the heuristic compared to the weight of the exact solution. We also showed for each weight interval the number of multicast requests in it. The results are shown in Fig. 7a.

What we first observe is that our heuristic always finds solutions of average weight at the most 10.5% greater than the best solution. We also observe that the performance of our heuristic decreases when the average weight of the multicast trees increases. This degradation can be explained as follows. An average greater weight of the multicast tree is due to either more destinations or destinations farther from the multicast source (or both). When the heuristic constructs the multicast tree, it tries in priority to add destinations to the diffusing nodes already selected in the partial solution because the weight of the arcs already used is set to zero (Section VII). When the number of destinations is not large enough to select most of the diffusing nodes in the solution, the set of diffusing nodes selected by the heuristic is most often different than the set selected by the exact algorithm and the heuristic solution diverges more from the best solution when the average weight increases. It is only after all or most of the diffusing nodes have been selected in the partial solution, does the heuristic add each new destination in the best way. To observe the performance of our heuristic when the number of destinations is greater, we generated more multicast requests by using another distribution $N_2 = \mathcal{N}(25\%\#T, 2\%\#T)$ to fix the number of destinations. The results are shown in Fig. 7b. We observe that the heuristic

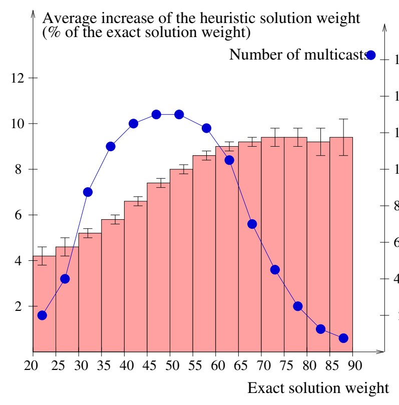

Fig. 7a

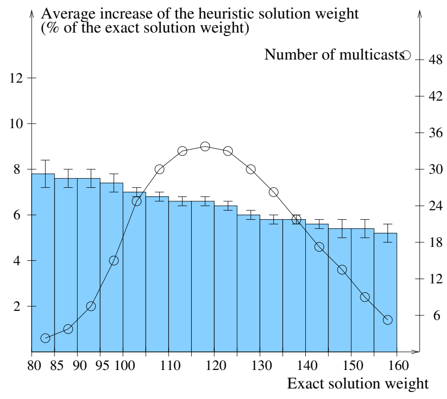

Fig. 7b

Fig. 7. Distribution of the multicast trees depending on their weight and the performance of the heuristic compared to the exact algorithm for a network with 200 nodes. The number of destinations is generated with N_1 (Fig. 7a) and N_2 (Fig. 7b). Confidence intervals computed with the significance coefficient $\alpha = 0.05$.

constructs solutions whose weight is closer to the best one. In cases in which the number of destinations is relatively large, both the exact solution and the heuristic one are constructed with all or most diffusing nodes. After all diffusing nodes have been selected by the heuristic, the heuristic adds each new destination in the best way and the weight of its solution becomes closer to the best one.

In order to observe the influence of the network size on the performance of our heuristic, we generated topologies of $i * 100$, $i = 1, 2, ..., 5$ nodes preserving the same average degree of nodes. We placed a various number of diffusing nodes ($\#D_G = 6, 8, 10, 12$) in each network using the k-center algorithm. As above, for each multicast source we generated 30 different destination sets whose cardinality is given by the distribution N_1. We computed the number of links used by the multicast trees with both our heuristic and the exact algorithm (Fig. 8a).

We observe that for a fixed number of diffusing nodes our heuristic performs better for greater networks. We have shown in Section VI that the non-polynomial part of the exact algorithm corresponds to finding the set of diffusing nodes used in the minimal multicast tree. When the set of diffusing nodes of the multicast tree constructed by the heuristic is almost the same as the set of diffusing nodes of the best multicast tree, the heuristic finds a multicast tree whose weight is close to the weight of the best multicast tree. In our model, since we always preserve the average degree of nodes and use the distribution N_1 based on the number of nodes of low degree, the average number of destinations depends on the network size. When the network size becomes larger, the average number of destinations for the generated multicast groups increases and more of the diffusing nodes must be used to construct the best multicast tree. All or most of the diffusing nodes are chosen by both the exact algorithm and the

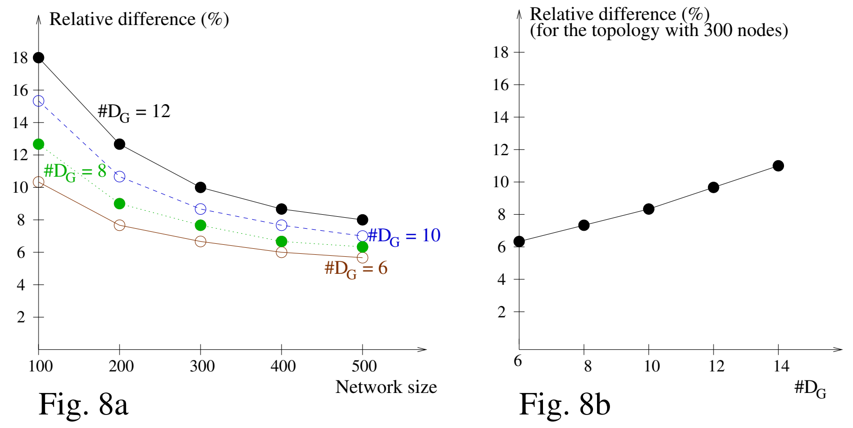

Fig. 8. Relative difference between average sizes of multicast trees obtained on one hand with the heuristic and on the other with the exact algorithm depending on the network size (Fig. 8a). Fig. 8b shows the same relative difference, depending on the number of diffusing nodes in the network with 300 nodes. Confidence intervals for both Fig. 8a and Fig. 8b are computed with precision 5% and significance coefficient $\alpha = 0.05$.

heuristic. The heuristic then constructs a multicast tree whose weight is close to that of the best one.

The results depending on the number of diffusing nodes in a given network (Fig. 8b) tends to confirm that for this given network, increasing the number of diffusing nodes degrades the performance of our heuristic. For the reasons explained above, the heuristic constructs better solutions when the ratio between the number of diffusing nodes and the number of destinations is small. A greater ratio means that a smaller portion of the diffusing nodes appears in the best multicast tree and the heuristic choses the same set of diffusing nodes less often than in the previous experiments.

9 Conclusion and Perspectives

We studied a construction of a multicast tree optimizing bandwidth in an optical MPLS network. We proved that the construction of a multicast tree optimizing the number of links used is NP-complete when only a subset of the routers are diffusing nodes. We proposed a heuristic algorithm to construct the multicast tree optimizing the number of links used and we evaluated the performance of our heuristic with a pseudo polynomial algorithm. We showed that the performance of the heuristic depends on the ratio between the number of diffusing nodes and the number of destinations. We observed that our heuristic performs very well when this ratio is small and that it constructs multicast trees whose weight is close to the weight of the best ones. We are currently working on new heuristics to construct a multicast tree and we will compare them with the heuristic we proposed here. We are also examining how we can improve the placement of the diffusing nodes in the network when we are given the set of the multicast sources.

References

1. The CARRIOCAS website, http://www.carriocas.org
2. D. Verchere. Orchestrating optimally IT and network resource allocations for stringent distributed applications over ultrahigh bit rate transmission networks, ECOC 2007.
3. O. Audouin. CARRIOCAS description and how it will require changes in the network to support Grids. OGF, 2007.
4. V. Reinhard, J. Tomasik. A mechanism of application-network interactions in a high-speed optical network. CISIS 2008.
5. X. Zhang, J. Wei and C. Qiao. Constrained Multicast Routing in WDM Networks with Sparse Light Splitting. INFOCOM 2000.
6. H. F. Salama, D. S. Reeves and Y. Viniotis. Evaluation of Multicast Routing Algorithms forReal-Time Communication on High-Speed Networks. IEEE J. on Sel. Areas in Comm. , Vol. 15, No. 3, April 1997.
7. J. Myoungki, X. Yijun, H.C. Cankaya, M. Vandenhoute, C. Qiao. Efficient multicast schemes for optical burst-switched WDM networks. IEEE ICC 2000.
8. R. Malli, X. Zhang, and C. Qiao. Benefits of multicasting in all-optical networks. In SPIE Proceedings, All Optical Networking, pages 209-220, November 1998.
9. J. Beasly. An SST-based algorithm for the Steiner problem in graphs. Networks, 19:1-16, 1989.
10. F. Hwang and D. Richards. Steiner tree problems. Networks, 22:55-89, 1992.
11. V. Kompella, J. Pasquale, and G. Polyzos. Multicasting for multimedia applications. INFOCOM 1992.
12. K. Bharath-Kumar and Jaffe. Routing to multiple destinations in computer networks. IEEE Transactions on Communications, COM-31(3):343-351, March 1983.
13. A. Medina, A. Lakhina, I. Matta, and J. Byers. BRITE : An Approach to Universal Topology Generation. MASCOTS 2001.
14. E. Mannie, Ed. Generalized Multi-Protocol Label Switching (GMPLS) Architecture, IETF RFC 3945, Ocotber 2004.
15. E. Rosen,A. Viswanathan, R. Callon. Multiprotocol label switching architecture, IETF RFC 3031, January 2001.
16. J. Mihelic and B. Robic. Solving the k-center Problem Efficiently with a Dominating Set Algorithm. J. of Comp. and Info. Tech. - CIT 13, 2005, 3, 225-233.
17. R. M. Karp. Reducibility Among Combinatorial Problems. In Complexity of Computer Computations, 1972. New York: Plenum, 85-103.