Choosing the Set of Rendezvous Points in Shared Trees Minimizing Traffic Concentration

Francesc Font, Daniel Mlynek
Signal Processing Laboratory - LTS3 - ITS
EPFL - Ecole Polytecnique Federale de Lausanne
1015 Lausanne, Switzerland
francesc.font@epfl.ch

Abstract[1] ***- In the current Internet there are two main types of intra-domain multicast routing protocols: dense mode and sparse mode. Dense mode protocols construct shortest path trees from a sender to receivers while sparse mode protocols construct shared trees rooted at a certain router, called core or RP (rendezvous point), to which all senders of a certain group send their packets in order to be transmitted to all receivers along such shared tree. These kinds of trees are well suited for wide-area networks, because we are sending information only to interested receivers and we maintain the same routing state for each group, independently of the number of senders, and only on the routers along the shared tree. While solving important scalability problems, such trees bring other drawbacks; mainly traffic concentration and delay. In current standards, PIM-SM[1] and CBT[2], no algorithms are specified to choose such RPs. In this paper we present an approach to select the best set of RPs that minimizes traffic concentration while limiting the maximum delay.***

I. INTRODUCTION

With the appearance of new applications, like TV over Internet, videoconferencing, distributed interactive simulation (DIS), multi-player gaming, interactive distance learning, etc, point-to-multipoint and multipoint-to-multipoint connections have become a necessity. To avoid sending repeated information to each receiver, with the consequent waste of bandwidth, the source sends unique multicast packets that are replicated downstream by routers when they find non-common paths for the different receivers. Some protocols have appeared to support these kind of connections, depending on the distribution, sparse or dense, of the receivers. The dense-mode protocols use specific delivery trees from the source to the receivers, while the sparse-mode protocols concentrate the traffic in certain RPs (rendezvous points), one per group, from which one tree for each group is built.

Dense-mode protocols, like DVRMP[3] and PIM-DM[4], use a *flood and prune* algorithm to find where the receivers are located and to construct the direct shortest path from the sender to each of the receivers (sender-oriented algorithm). The state for each (Source, Group) pair is maintained in all routers, independent of whether they are placed or not along the shortest path to receivers. It allows a new receiver to join (graft) the shortest path tree for a certain (S, G) pair at any time. Otherwise, MOSPF[5] avoids such a flooding by maintaining a complete map of where the receivers are in each router. Such protocols are suitable for single-sender groups where the receivers are densely distributed along the domain. But this protocol architecture presents serious scalability problems with the number of senders and groups especially in WANs, where the receivers are sparsely distributed along the domain.

In order to improve such scalability problems sparse-mode protocols use the concept of rendezvous point. An RP is a router that acts as a meeting point between all the senders and receivers of the same group. Thus, one shared tree is built from the RP to all the receivers, allocating group state information only in the routers along the shortest path from each receiver to the RP, that means, along the reverse path. The same tree is valid for all the senders, because all the sources of the same group send the information to the same RP. From the RP, the information is disseminated to the receivers along the shared tree. The group-to-RP mapping is flooded to the entire domain using a Bootstrap algorithm[6], that allows all the DRs, designated routers[2], mapping all members of a group to the same RP.

While bringing advantages in scalability issues, sparse mode protocols have also some drawbacks as traffic concentration around RPs and longer delays than in dense mode [7](triangular routing problem). Besides, PIM-SM and CBT, do not specify any criteria to choose the RPs.

In this paper we present an approach to assign priorities to the different C-RPs, candidates to became rendezvous point, in order to make PIM-SM choose the set of RPs minimizing traf-

[1] This work is done in collaboration with TranSwitch SA.

[2] Routers with attached hosts

fic concentration while limiting maximal delay. The rest of the paper is organized as follows. In section II we present the state-of-the-art in core selection methods. In section III we show our algorithm and in section IV. we state some demonstrations. In section IV we present an example and we explain results of simulations. Finally, we present some conclusions and some highlights for further investigation in section IV.

II. CORE SELECTION METHODS

The currently implemented shared-tree protocols, CBT and PIM-SM do not use an efficient method to chose the core (RP) for each group [8]. A RP is pseudo-randomly assigned to each group using a hash function that ensures that no more than four consecutive groups will map to the same RP. Such a method avoids an unequal distribution of groups between all the RPs when the number of groups is large. The hash function is also designed to bring the minimal disruption of groups (changes in the group-to-RP mapping) when there is a change in the set of C-RPs[6]. The C-RPs are assigned administratively and fight to become an RP for a certain pre-configured range of groups and with a pre-configured priority. Thus, the first question that arises is, what is the priority to be assigned to each router in order to optimize shared trees?

All the studies been made until now show how to choose the core (RP) in order to minimize the delay or even the link cost[9]. In orther to reach such goals migration algorithms have been created in order to "change" the RP assigned to a certain group when another RP fits better delay requirements[10][11][12][13][14]. The practical application of these methods depends on:

- The availability of the information about senders and receivers.
- Efficient and reliable methods of changing the core when the participant distribution changes.

Having information about group participants is difficult. Besides, changing the core depending on the group participants can bring undesirable instabilities and loss of data while changing the core[6]. All these drawbacks lead the currently implemented protocols (PIM-SM and CBT) to use random core selection methods.

III. MINIMIZING TRAFFIC CONCENTRATION

Currently, priorities of C-RPs are assigned by hand. Higher priorities are given to "best connected" routers in order to have a good distribution of traffic among links. The set of routers with the highest priorities is elected as the set of RPs. Using a hash function, each group pseudo-randomly maps to a certain router of such a set. The first question that arises with such an algorithm is what are the routers that give the best distribution of traffic among available links. Traffic concentration is not taken into account by group-based methods, where the core is placed in the topological center of the members of a group in order to minimize delay. When having widely distributed senders and receivers, such methods may bring us high traffic concentrations, that increase delay due to queueing[10]. As we may have a huge number of different groups, equally distributed in number among the entire set of RPs, we can assume "a priori" an equal distribution of traffic among RPs and an homogeneus distribution of senders and receivers of all groups mapping to a certain RP. In such conditions, we will show a method that will allow us to know the set of RPs minimizing traffic concentration. This method will assign priorities to the different C-RPs, improving the performance of shared tree methods, without any modification of the protocol (PIM-SM or CBT).

A The Algorithm

We can represent a network as a graph $G(V, E)$; where $|V|$ is the order of G (number of vertices or nodes) and $|E|$ is the size of G (number of edges or links). Such a graph (e.g. Fig. 1) can be entirely represented by a square matrix, the *Adjacency Matrix* A, where each component represents the existence "1" or inexistence "0" of an edge between two vertices. We may also have components bigger than 1, representing the weight[3] of the link [15][16].

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix}$$

For our algorithm we have designed two more matrices, the *Broadcast Matrix* and the *Combination Matrix*. Each row of the *Broadcast Matrix* represents the "generic" usage of links when choosing the corresponding vertex as a RP for certain groups having, globally, members distributed uniformally along the graph[4]. To get such vectors, we send one packet from the selected vertex to all the other vertex with attached

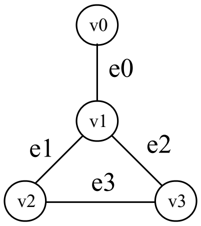


Fig. 1. Example of a graph with 4 vertices and 4 edges.

[3] Link cost

[4] The Graph represents a domain or autonomous system running PIM-SM or CBT

hosts, and we note the usage of each link. Thus, for the graph in Fig. 1, the *Broadcast Matrix* will be:

$$B = \begin{bmatrix} 3 & 1 & 1 & 0 \\ 1 & 1 & 1 & 0 \\ 1 & 2 & 0 & 1 \\ 1 & 0 & 2 & 1 \end{bmatrix}$$

where the rows represents the vertices *{v0, v1, v2, v3}* and the columns represents the edges *{e0, e1, e2, e3}*. We are assuming a link cost of "1" for all the links. For different link costs we only need to multiply the link cost of an edge to the components of the corresponding column. Similarly, we can "adapt" the components of the Broadcast Matrix to other parameters (g.e. bandwidth). From this matrix, we want to find the combination of rows minimizing the usage of links. Such an addition $a_0, a_1, \ldots, a_z$ of N nodes will accomplish:

$$M^n \times (a_0^n + a_1^n + \ldots + a_z^n) < N^n \times (b_0^n + b_1^n + \ldots + b_z^n) \qquad (1)$$

for all other combinations $b_0, b_1, \ldots, b_z$ of M nodes, where $M = 1, \ldots, |V|$, $z = |E|$ and n is the criteria to choose the best set; $n = 2$ to minimize the average usage of links and $n > 2$ to penalize high concentration in few links. As generating all the combinations is quite arduous for graphs with hundreds or nodes and gives as a result only the best combination of nodes, we have designed an heuristic to progressively generate local best sets and eliminate nodes; this heuristic is based on the fact that edges linking the analyzed subset of nodes will have the highest values, because we are broadcasting packets from such nodes in order to extract the components of their corresponding rows in the *Broadcast Matrix*. If we minimize the values for such links, we will also have a minimum when combining this subset with another one, minimizing another different subset of links. Thus, we reduce the computation cost while achieving the goal of classifying nodes in groups depending on "how good" they are in terms of link usage. To reach such a goal, we create the *Combination Matrix.* Each row of this matrix represents the best combination of the node corresponding to this row and all its neighbours; remember that such a neighbourhood is represented in the *Adjacency Matrix*.

Thus, for the graph in the Fig. 1, the *Combination Matrix* is:

$$C = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}$$

As for all the nodes, the best combination is v_1, this node is elected as the best set of RPs for this graph. We assign the next priority to v_0, v_2, v_3. In this case, we have found the solution in the first iteration. All-zero columns in the *Combination Matrix*, represent nodes to be eliminated. For bigger graphs, we need several iterations, eliminating in each one a certain set of nodes that is "labeled" with the same priority. In order to go on the next iteration, we will modify the *Adjacency Matrix*, representing the new generated neighbourhood, when not taking into account the eliminated nodes. From this new *Adjacency Matrix* and the *Broadcast Matrix* we are able to generate a new *Combination Matrix*. Again we eliminate the nodes corresponding to all-zero columns. Iterations finish when the *Combination Matrix has no all-zero columns.* Finally, we extract the best set of nodes from the remaining ones, assigning to them the highest priority. We assign to the others the next priority level. The pseudo-code of the algorithm is presented in Fig. 2.

IV. DEMONSTRATIONS

In this section, we present two special types of graphs demonstrating, for each of them, what are the best set of vertices to be elected as RPs.

A Radial-Tree graphs

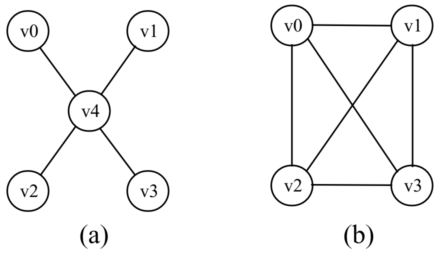
An example of radial tree is presented in Fig. 3(a). In this type of graph, only the central node is elected as the best set of RPs. The demonstration comes from the fact that nodes with only one link will be eliminated by the node to whiche they are attached. We can see such a situation in the *Broadcast Matrix* B_{rt}, where v_0 (first row) is a node with only one link attached to the node v_1(second row), m is the number of vertex and n is the number of edges. As v_1 has all the components lesser or equal than v_0, applying (1) v_0 will be

```
do
{
   for (i=0; i<num_vertex; i++)
   {
      //take into account vertex i and its neighbours
      extract_best_subset_of_nodes;
      write_it_as_a_row_in_Combination_Matrix;
   }

   eliminate_vertex; //assign priority
   generate_new_Adjacency_Matrix;

}while (all-zero-columns_in_Combination_Matrix);
extract_best_subset_of_nodes; //assign best priority
assign_next_priority_to_remaining_nodes;
```

Fig. 2. Algorithm to assign priorities to C-RPs



(a) (b)

Fig. 3. (a) radial-tree (b) complete graph K^4

eliminated in the first iteration, with the consequent assignation of the lowest priority.

$$B_{rt} = \begin{bmatrix} m-1 & v_0^1 & \dots & v_0^n \\ 1 & v_0^1 & \dots & v_0^n \\ \dots & \dots & \dots & \dots \\ v_m^0 & v_m^1 & \dots & v_m^n \end{bmatrix}$$

B Complete Graphs

An example of complete graph[16] is presented in Fig. 3(b). In this type of graph, all the nodes are directly connected to each other. Thus, in the *Broadcast Matrix,* it is accomplished that

$$\sum_{x=0}^{m} v_x^y = 2, \forall y \qquad (2)$$

where m is the number of rows and y the different columns. In such a situation, when aplying (1), all nodes will be elected as the best set of RPs.

C Other Graphs

Any other connected[5] graph[16] has characteristics “between” radial tree and complete graphs. Thus, any number of vertices between 1 and $|V|$ will be elected as the best set of RPs, for all connected graphs.

IV. SIMULATIONS

As an example we have generated a graph of 20 vertices with a Waxman distribution model, using an average node degree of 3 (Fig. 3), in order better reflect a realistic network structure[17]. All simulations using exponential factors ($n = 2, 3, 4$) give the same results (table I), representing vertices from lowest to highest priority. We have generated different graph structures with GT-ITM[6] (random, Waxman, transient-stub and hierarchical), with different number of vertices and node degrees, and run simulations with Network Simulator[7]. In all our simulations, the two highest priority vertices have an eccentricity[8] that accomplishes:

$$eccentricity \le (radius + diameter)/2 \qquad (3)$$

That means that even if recovering RP failures with second priority vertices, we will have a maximum delay between senders and receivers of $1 \cdot 5 \times RTT$. In Fig. 4, we differentiate four levels of centering. Level1 (topological centers[8] of the graph: 2, 6, 9, 10, 11, 13, 15, 18), level 2 (0, 5, 14, 16, 17), level 3 (1,3,8) and level 4 (peripheral nodes: 4, 7, 12, 19). In our example, nodes with highest priority are all in level 1 (all are centers of the graph), and nodes with the next priority are in levels 1 and 2.

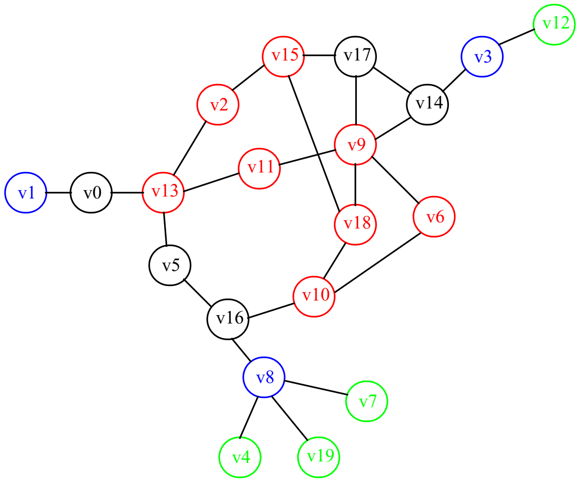


Fig. 4. Waxman graph with 20 nodes

IV. CONCLUSIONS AND FURTHER STUDY

In this paper, we have presented an algorithm to systematically assign priorities to C-RPs in a PIM-SM domain in order to minimize traffic concentration in links, also limiting the maximum delay to $1 \cdot 5 \times RTT$, even when having RP failures. This gives us a good criterion to compare PIM-SM with other multicast routing protocols.

In PIM-SM, priorities are pre-configured in routers, and asymetries in the traffic mapped to each RP or non-uniformity in group members’ distribution cannot be efficiently managed by PIM-SM. Core migration algorithms also bring some problems, as we have explained in section II. Our future work will

[5] Any two vertices are linked by a path in G

[6] Georgia Technology Internetwork Topology Models http://www.cc.gatech.edu/fac/Ellen.Zegura/graphs.html

[7] http://www-mash.cs.berkeley.edu/ns/

Table I: Assigned priorities to nodes of the Waxman graph

Priority	Vertices
0	1, 4, 6, 7, 12, 19
1	0, 3, 8
2	14
3	2, 5, 10, 11, 16, 17
4	9, 13, 15, 18

be focused in the use of several RPs per group, in order to better distribute traffic along the network and better assume non-uniform distribution of senders and receivers.

ACKNOWLEDGMENT

We would like to acknowledge useful discussions about the algorithm with Brice Tsakam and text revisions of Marc Epalza, both of TranSwitch S.A.

REFERENCES

[1] D.Estrin, D.Farinacci, A.Helmy, D.Thaler, S.Deering, M.Handley, V.Jacobson, C.Liu, P.Sharma and L.Wei. "Protocol Independent Multicast-Sparse Mode (PIM-SM): Protocol Specification". RFC 2362, June 1998.

[2] A. Ballardie. "Core Based Trees (CBT version 2) Multicast Routing". RFC 2189, September 1997.

[3] D. Waitzman, C. Partridge and S. Deering. "Distance Vector Multicast Routing Protocol". RFC 1075, November 1988.

[4] S. Deering, D. Estrin, D. Farinacci, V. Jacobson, A. Helmy, D. Meyer and L. Wei. "Protocol Independent Multicast version 2 Dense Mode Specification". draft-ietf-idmr-pim-dm-06.txt, Aug 6, 1997.

[5] J. Moy. "Multicast Extensions to OSPF". RFC 1584, March 1994.

[6] D.Estrin, M.Handley, A.Helmey, P.Huang and D.Thaler. "A Dynamic Bootstrap Mechanism for Rendezvous-based Multicast Routing". Technichal report, February 1998.

[7] L.Blazevic and J.Y. Le Boudec. "Distributed core multicast: a multicast routing protocol for many groups with few receivers". Proceedings of First InternationalWorkshop on Networked Group Communication, Pisa, November 1999.

[8] H.F.Salama, D.S.Reeves and Y.Viniotis. "Shared Multicast Trees And The Center Selection Problem: A Survey". ECE Department and CSC Department, North Carolina State University. http://rtcomm.csc.ncsu.edu/Center-SectionSurvey.pdf

[9] L.Wei and D.Estrin. "The trade-offs of multicast trees and algorithms". In Proceedings of ICCCN ’94, 1994.

[10] K.Calvert and E.Zegura, M.Donahoo. "Core Selection Methods for Multicast Routing". In Proceedings of IC3N ’95, 1995.

[11] S. Ali, A. Khokhar. “Distributed Center Location Algorithm for Fault-Tolerant Multicast in Wide-Area Networks “. Symposium on Reliable Distributed Systems, 1998.

[12] E. Fleury, Y. Huangand, P. K. McKinley. “On the Performance and Feasibility of Multicast Core Selection Heuristics”. In Seventh International Conference on Computer Communications and Networks, Lafayette, Louisiana, October 1998.

[13] Y. Lin, N. Hsu, C. Pan. “Extension of RP Relocation to PIM-SM Multicast Routing”. IEEE International Conference on Communications, Helsinki, June 2001.

[14] Y. Lin, N. Hsu, R. Hwang. “RP Relocation Extension to PIM-SM Multicast Routing”. IETF Internet-Draft, draft-ydlin-pim-sm-rp-00.txt, April 2001. Expires April 22, 2002.

[15] G. Chartrand, Western Michigan University. “Introductory Graph Theory”. Dover Publications, Inc. New York, 1977.

[16] R. Diestel, University of Hamburg. “Graph Theory“, Second Edition. Springer-Verlang New York 1997, 2000.

[17] E.W.Zegura, K.Calvert, S.Bhattacharjee. "How to Model an Internetwork". In Procedure of INFOCOM’96, 1996.