Mobile Agents based QoS Multicast Routing (MAQMR)

Mohamed EL HACHIMI , Abdelhafid ABOUAISSA, Pascal LORENZ
Haute Alsace University, 38, Rue Grillenbreit, 68000 Colmar, France

Abstract

Many multicast applications, such as video-on-demand and tele-education, desire quality of service(QoS) support from the underlying network. Recently, many QoS-based multicast protocols have been proposed to meet these requirements. However, the process of changing and deploying theses protocols are lengthy and difficult, because the process require standardisation and automatic mechanism for dispatching programs for new protocols. This paper describe a new agents based multicast QoS routing protocol (MAQMR). This approach, first, allows the multicast protocol to be implemented within mobile agents and dynamically deployed at every node the agent is visiting. Second, independently of unicast routing, mobile agents act in parallel and distributed fashion cooperate in order to construct the multicast tree while minimizing message overhead and satisfying QoS requirements.

INTRODUCTION

The phenomenal growth of group communications and quality of service (QoS)-aware applications over the Internet has accelerated the need for scalable and efficient network support. Several multicast routing protocols have been proposed in the literature with varying performance, cost, and implementation [7][8]. However, few of them can achieve high success ratios while keeping good scalability. In addition current process of changing and deploying network protocols are both lengthy and difficult. The process often requires standardization, which takes some years to be consented. Furthermore, once the new protocols have been accepted, their deployment is difficult, because there is not any automatic mechanism for dispatching programs.

To counter the above problems, a novel and powerful scheme is introduced in this article that provides a means of supporting multicast QoS routing through the use of mobile agents. A software agent is loosely defined as a program that can work autonomously toward a goal, and meet an interact with other agents and its environment. It comprise the code and state information needed to carry out some computation. Agents may be static or mobile. Static agents remain resident at a single platform, while mobile agents are capable to change the platform [].

Mobile software agents provide a new and useful paradigm for distributed computing. Specifically, we describe the use of mobile agents to efficiently realize point-to-multipoint routing trees, while satisfying the QoS requirements. In this paper, we propose a mobile agents based protocol to support multicast or group communications and provides QoS-sensitive paths in a scalable, resource-efficient and flexible way. The proposed protocol MAQMR identifies multiple paths satisfying QoS requirements and select the best one in terms of cost (hop count). It achieves scalability by significantly reducing the communication overhead of constructing a multicast tree, and optimise resources by selecting the best QoS compliant path to connect the new receiver to the multicast tree. Therefore, it doesn't require global link-state information neither global topology knowledge.

The use of mobile agents allow our protocol to act in distributed fashion, since the QoS compliant path should be decided by a cooperative and parallel task. This increase dependability of the network by avoiding point of failure of a centralized network management system. The multicast QoS routing protocol (MAQMR) doesn't depend on the unicast routing protocol since the mobile agent carry the list of visited nodes and current link states. It is also platform independent since the protocol is implemented in the mobile agent instead of routers. The rest of this paper is organized as follows. Section 2 describe related works and motivations, in section 3 we detail the proposed protocol and present the algorithm used. In Section 4, we present a performance study and simulation results. Finally, conclusions and future work are presented.

1. The Network Model

As far as multicast routing is concerned, a network is usually represented as a weighted digraph $G = (V, E)$, where V denotes the set of nodes and E the set of communication links connecting the nodes. Associated with each link are parameters that describe the current status of the link. For example, a link-delay function which assigns the delay that packets experience on link $l \in E$, and takes into account the queuing delay, transmission time, and propagation delay. Another example is the bandwidth available on an outgoing interface as a link parameter. These parameters are collectively termed *link stat*e, and are usually maintained by a node. Let $M \subset V$ be a set of nodes involved in a group communication. We call set M a multicast group with each node $v \in M$ a group member. Packets originating from a source node v_s have to be delivered to a set of receiver nodes $M - \{v_s\}$. A multicast tree T is a sub-graph of G that spans all the nodes in M.

Given a multicast group M, multicast routing is the process of constructing, based on network topology and network state, a multicast tree T that minimize the cost of a multicast tree. The resulting multicast tree must provide not only reachability from source(s) to a set of destinations, but also certain QoS merits on the routes found in order to satisfy the constraints. In our protocol the constraints we are taking into account are bandwidth and delay. The cost of the path is in term of hop count. We define a feasible path as a path that satisfy QoS requirement.

2. Related work

In this section, we present some background on QoS and multicasting in the Internet environment. We classify and review briefly a number of protocols and highlight the needs that motivate our work.

Recently, several QoS multicast routing algorithms have been pro-posed to find feasible trees. Some algorithms provide heuristic solutions to the NP-complete *constrained Steiner tree proble*m, which is to find the delay-constrained least-cost multicast trees. These algorithms, however, are not practical in the Internet environment because they have excessive computation overhead, require knowledge about the global network state, and do not handle dynamic group membership. The spanning join protocol by Carlberg and Crowcroft [3] handles dynamic membership and does not require any global network state. However, it has excessive communication overhead because it relies on flooding to find a feasible tree branch to connect a new member.

In QoSMIC [4], the search for candidate paths consists of two parallel procedures: local search and tree search. The local search is equivalent to spanning join, except that only a small neighbourhood is searched. The tree search handles the case when there is no on-tree node in the neighbourhood checked by local search. In the tree search, the new member contacts a designated manager node that is responsible for ordering a subset or on-tree nodes to establish a path from them to the member. Each such path is a candidate path. The new member then selects the best path out of these candidate paths. This protocol is alleviates but does not eliminate the flooding behavior. In addition, an extra control element, called Manager router, is introduced to handle the join requests of new members.

The QoS-aware multicast routing protocol (QMRP) [5] consists of two sequential procedures: single-path mode and multiple path mode. The protocol starts and continues with single-path mode until it reaches a node that has insufficient resources to satisfy the join request. When such a node is encountered, the protocol switches to multi-path mode. It constructs a shared tree by unicasting a request message from the host router toward the core router (or source router). If a router in the unicast path does not satisfy the QoS requirements, the request message is replicated and sent out to all other neighbours of the router. It introduces the idea of QoS-awareness into the path selection period, which increases the ability of finding a feasible branch. However, it requires temporal state in the network routers for each join request. It is only applicable for applications with non-additive QoS requirements such as bandwidth and buffer space, and cannot be used for additive requirements such as delay or packet loss.

QMBF [6], utilizes the edge nodes' least-cost information and the M-hop bounded flooding method to control the flooding traffic. This protocol two steps to find a feasible path. First is local searching. The node search for edge nodes which have the least-cost toward the target router. Second, it computes a feasible path from itself to the selected edge using LNC information, and sends the join message and reserves resources along the path. However, it depend on unicast routing protocol and requires every node periodically broadcast its local QoS state information and unicast reachable information.

3. Protocol overview

The multicast routing protocol presented here is based on a colony of mobile agents deployed for the actual discovery of QoS-compliant routes. The benefits of this scheme can readily be seen. Compared to classical routing, there is no need to perform a whole routing table re-computation to update the QoS information at remote locations each and every time there is a change in the state of the individual router. Therefore, the need to keep larger routing tables holding information on all QoS-related parameters is eliminated, since the mobile agent discover dynamically link state information during its travel. Moreover, there is no need to flood the network with routing tables to and from every network node, either periodically or triggered by significant changes.

3. Detail description

3.1 QoS Path Finding using Mobile Agents

The process of adding a new branch to the multicast tree starts when an edge router receives a request to join a multicast group. If the edge router is part of the group already, the connection is established locally. If the edge router is not part of the group, feasible path searching algorithm is employed to connects the new member to the available tree.

In the proposed protocol (MAQMR), feasible path searching algorithm consists to use a coordinated colony of mobile agents launched from the edge router. The proposed algorithm is an adapted version of the algorithm proposed in [10], where a similar Mobile agent based approach is used to establish **Multipoint to Point Tree.** In our algorithm we establish a **Point to Multipoint Tree** (multicast tree). Main differences are: in [10], the path discovery step consist to **two sets of agents**, first set of agents is launched from each edge router and allowed to travel if they have travelled a **shorter distance** than one previously recorded by a different agent coming from the same origin. In addition agents from this set doesn't record visited nodes it is done only in the second set of agents. The second of agents set is similarly launched and allowed to continue their traversing toward the destination if the current travelled **distance equals** that recorded by the previous set of agents. In our algorithm **only one set of agents** is sent and they continue their traversing if the current travelled distance **less or equals** than one previously recorded by a different agent coming from the same origin. In this step agents record visited nodes. The other difference consist of using a variable called "visits" in [10], this variable is used in each node to assert visit of agents coming from different origin nodes. Then, a third set of agents is launched to update this variable. In our algorithm we don't use this variable and we don't launch no set of agents to update it. The final difference is that in [10] the decision to select between two parallel paths is based on the "weight" the path is credited, which is related to the "visit" variable. In our algorithm the selection is based on how the path is long which is related to the visited nodes recorded. Following rules are used to describe the comportment of a mobile agent.

Rule 1. An agents clone its self only on links whose unreserved resource numbers are not less than the bandwidth required. [10]

Rule 2. An agent is allowed to continue its travel if the distance carried is less [10] or equal than one previously recorded (this is applied for agents coming from the same origin and looking for the same multicast group).

Rule 3. If an agent visits the same node at least twice, then the agent die.

The agent is allowed to travel only on links that meet the QoS constraint required. First, this allow to find only QoS compliant routes and second, it reduce the number of mobile agents. When an agent reach a node, the agent clone its self as many copies as QoS outgoing links the node has. An agent during its travel carry a list of visited nodes and cost local variable (Agent_distance_travelled). This variable is incremented arriving in an intermediate node. The cost value is used by each node to allow any agent having travelled the minimum partial distance than one previously recorded by a different agent coming from the same origin node and looking for the same multicast tree to continue its travel toward the destination. An agent carrying a larger distance is discarded at any intermediate node as soon as this condition is detected. The distance here is hop count and could be any other function. This process will reduce the overhead created by mobile agents.

Two important results in this process are first, all possible QoS paths are founded and the second is that the routes found are cycle free, since if an agent return to an earlier visited node will carry a cost larger than the previous cost recorded during its first visit to the same node. Therefore, it will be discarded (Agent_state = dies). A mobile agent stop to be cloned even reach the source of the multicast tree or reach an intermediate on tree node for the group the agent is looking for. In this case the agent doesn't clone itself, change its state (Agent_state = Backtracking) and travel back to its home node (Agent_origin_node) flowing back the founded path.

Finally, the home node will contain a list of all shortest path from itself to the multicast tree, that comply with QoS constraint required. Then a selection process allow to find the best QoS compliant route using the cost carried by the agents. This procedure allow to optimise the cost of the multicast tree. Then, the agent carrying the best cost (Agent_distance_travelled) changes its state (Agent_state = Constructing) and establish the path.

Figure 5 a) shows agents in searching state for QoS paths between (e_c) and the multicast tree. As explained previously in this section, mobile agents moves, dies, clone themselves and cooperates in order to achieve this goal. We can see that no agent moves through links ($c_k - c_m$ and $c_n - c_m$) because theses links doesn't meet receiver QoS requirements. In Figure 1 b) agents arriving in a multicast source (e_a) or an on tree node (c_n and e_d) travel back to (e_c). since all paths founded meet the new receiver's QoS requirements, (e_c) select the best one in terms of cost.

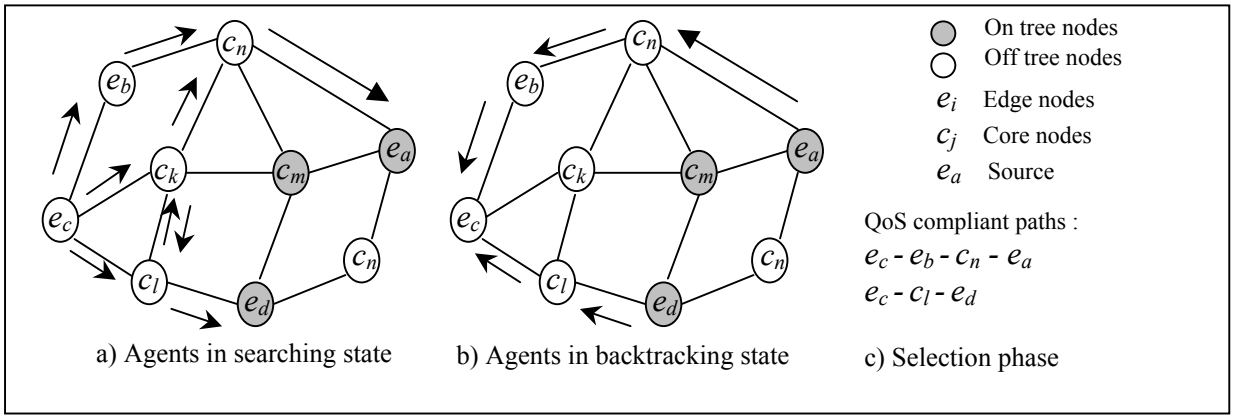

Figure. 1. Procedure for finding all possible QoS paths between a receiver and a multicast tree

3.2 Algorithm

```
Agent_visit_list = NULL
Agent_origin_node = origin
Agent_mcast_group = group
Agent_required_QoS = QoS
Agent_distance_travelled = 0

Agent_state = Searching
Repeat until (mcast-src or on-tree node) reached
{
Record current node in Agent_visit_list
Increment Agent_distance_travelled
Clone Agent and hop through all QoS-compliant links

If first Agent  reach a node,
Record Agent_origin_node, Agent_mcast_group and
Agent_distance_travelled

Else

If Agent_distance_travelled  <= previous recoded
{
If Agent_distance_travelled < previous recoded
{Update new distance for respective origin}

continue navigation
}
Else Agent_state = dies
}

Agent_state = Backtracking
reverse Agent_visit_list
Repeat until reversed Agent_visit_list is empty
{
get node from reversed Agent_visit_list and move to it
}

Agent_state = Constructing
Repeat until Agent_visit_list is empty
{
add-mfc(src,group)
get node from Agent_visit_list and move to it
}
```

3.3 Connection Establishment

We will examine how the new path is established after the searching phase has been completed.

1. Since all paths are QoS compliant, the edge router selects the best candidate according to the cost collected by the set of backward agents.
2. The edge router sends back the agent to the candidate who have been selected (selection phase Figure 1). The constructing agent traverses the path in the opposite direction, and establish routing state along the selected path.

3. When the chosen candidate (source or on tree router) receives the constructing agent, it starts transmitting data packets on the newly set-up path towards the edge router.

3.4 Leaving a Group

A Designated router receives a leave request through the same protocol that communicated the join request. The request can be for a whole group or for a source of a group. Whenever a router senses a change in the membership (request from a host, an agent "Prune" from a neighbouring router, it removes the link from the distribution tree, and checks whether it has become a leaf of the related tree. If so, it sends an agent "Prune" up the tree and removes the state for the tree from its database, thereby ceasing to be an In-tree router for that tree.

4. Simulation Results

4.1 Random graph generation

To ensure that simulation of the effects of different routing algorithms are fairly evaluated, random graphs with low average degrees are constructed. The nodes are randomly connected with the probability function:

$$P(u,v)=\lambda \exp(\frac{-d(u,v)}{\rho L})$$

where $d(u,v)$ is the distance between node u and v and L is the maximum possible distance between any pair of nodes. The parameters λ and ρ ranging (0,1] can be modified to create the desired network model. For example, a large value for λ gives nodes with a high average degree, and a small value for ρ increases the density of shorter links relative to longer ones. In our simulation, λ and ρ are set to 0.25 and 0.2, respectively to simulate large network such as Internet. We use the bandwidth of link between node u and v as the cost of the edge. The bandwidth capacity of each edge is randomly generated in]0,10]Mb.

4.2 Performance analysis

The algorithm previously described has been implemented using NS2. A dynamic scheme was created and implemented to simulate a number of random receivers requesting to either join or leave existing multicast tree. Based on above topology generation method, a network topology (size 100 nodes) is generated. For each simulation, a source and a set of 10 multicast receivers are randomly generated. Receiver's QoS requirement are fixed for each link success ratio (what percent of links meet the new receiver QoS requirements). For each data point plotted we run the simulation 100 times. We have mainly focused on success ratio per join as measure of performance.

4.2.1. Routing Efficiency

In this section, we evaluate MAQMR routing efficiency, measuring the success ratio per join defined as follow:

$$success\ ratio=\frac{number\ of\ new\ members\ accepted}{total\ number\ of\ join\ requests}$$

Figure 2 compares the success ratio per join of different protocols MAQMR, QMBF and QMRP. The figure shows that MAQMR provide a success ratio better than QMBF23 and QMRP2.

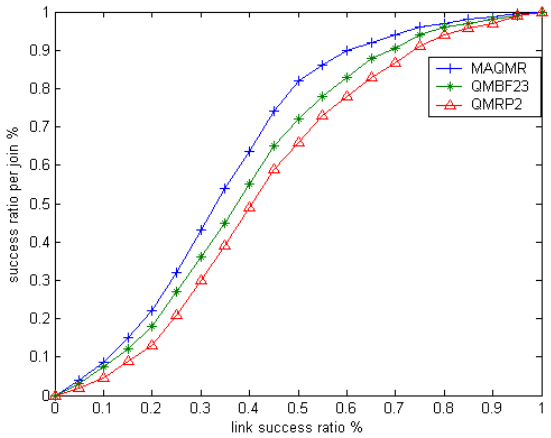

Figure 2 : Success ratio per join of different multicast protocols with multicast group size 10

Figure 3 provide a case study on how the maximum branching degree (MBD) affects the performance of MAQMR. As shown in Figure 3, a larger MBD results in a better success ratio and consequently introduce a larger overhead. Then, a good trade-off between success ratio and the agents overhead would be required. This concept of MBD could be used in the algorithm (section 3.2) in order to optimise the overhead.

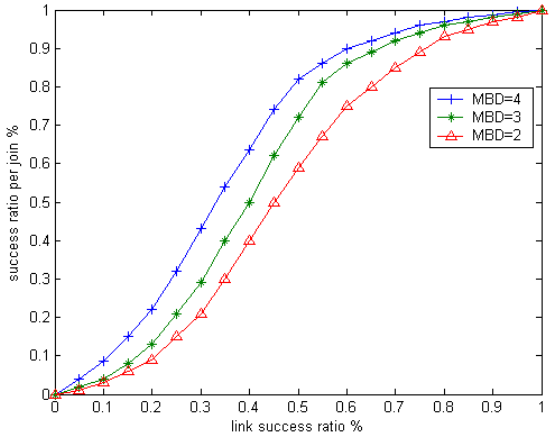

Figure 3 : Success ratio per join of MAQMR with multicast group size 10 and different MBD

Conclusion

In this paper, we propose a new QoS-aware multicast routing protocol (MAQMR). Basing on mobile agents, the multicast protocol can be dynamically deployed. Mobile agents works in parallel and cooperate in order to find a feasible path from the new member to the multicast tree. The protocol doesn't depend on the unicast routing protocol, and requires no intermediate routers to exchange links state information. The simulations results shows that MAQMR can achieve better success ratio than other QoS-based multicast routing protocols.

Future work will focus on how to reduce the number of agents in the network and the bandwidth consumed by them, more efficient techniques have to be investigated and tested. Complex methods might imply increased computational complexity of the agent algorithms. A deeper investigation and trade-off analysis would be required to establish an adequate equilibrium between these important factors.

Bibliography

[1] G. Di Caro and M. Dorigo. *"Mobile agents for adaptive routing*". In Proceedings of the 31st International Conference on System Sciences (HICSS-31), volume 7, pages 74--83. IEEE Computer Society Press, 1998.

[2] K. Oida and M. Sekido, "*ARS: An Efficient Agent-Based Routing System for QoS Guarantees,*" *Comp. Commun*., vol. 23, 2000, pp. 1437–47.

[3] T. Ballardie, P. Francis, and J. Crowcroft, "*Core-Based Trees (CBT): An Architecture for Scalable Interdomain Multicast Routing*," *Proc. ACM SIGCOM*M, 1993, pp. 85–95.

[4] M.Faloutsos, A. Banerjea, and R. Pankaj, "*QoSMIC: quality of service sensitive multicast internet protocol*," in *Proc. SIGCOMM'98*, sept 1998.

[5] Shigang Chen, Klara Nahrstedt, and Yuval Shavitt. "*A QoS-aware multicast routing protocol". IEEE Journal on Selected Areas in Communications,* Volume: 18 Issue: 12, Dec. 2000, pp. 2580-2592

[6] Z. Li and P. Mohapatra. "*QoS-aware Multicast Protocol Using Bounded Flooding (QMBF) Technique*". *In ICC*, May 2002.

[7] J. Hou and B. Wang, "*Multicast Routing and its QoS Extension: Problems, Algorithms, and Protocols*," *IEEE Networ*k, Jan./Feb. 2000.

[8] G. Manimaran, A. Striegel, "*A Survey of QoS Multicasting Issue*," *IEEE Networ*k, June. 2002.

[9] J. Hou and B. Wang, "*Multicast Routing and its QoS Extension: Problems, Algorithms, and Protocols*," *IEEE Networ*k, Jan./Feb. 2000.

[10] S. Gonzalez and V.Leung "*QoS Routing for MPLS Networks Employing Mobile Agents," IEEE Network*, May/June 2002.

[11] Network simulator, http://www.isi.edu/nsnam/ns