Applied Mathematical Sciences, Vol. 1, 2007, no. 57, 2839 - 2849

QoS-Constrained Core Selection Algorithm for Multicast Routing

Lin Huang[1], **Haishan Han**[2] **and Yulin Zhang**[3]

[1]Department of Applied Mathematics
Dalian University of Technology, China

[2]College of Mathematics and Computer
Inner Mongolia University for Nationalities, China

[3]College of Information and Engineering
Shandong University of Science and Technology

Abstract

With the proliferation of multimedia group applications, multicast is becoming increasingly important. We propose a core selection algorithm to the core selection problem in Core-based multicast routing with QOS requirements.we select the smaller set of cores from the set of candidate cores such that the number of group members with satisfied end-to-end QoS requirements is maximized. Simulation results show our algorithms have features of well performance of less selected cores,and are feasible and effective.

Mathematics Subject Classification: 68T20

Keywords: Quality of Service, Multicast routing,Core selection algorithm

1 Introduction

Multicast communication, the delivery of a data stream from a single source to multiple destinations in a computer network, has received a great deal of attention in recent years. Applications that use multicast communication include teleconferencing, computer-supported cooperative work, distributed interactive simulation, distance education, distributed operating systems, and distributed parallel processing. The growing importance of multicast communication in computer networking and telecommunications is demonstrated by the development and widespread use of the IP multicast and by the inclusion of multicast services in standards for the ATM,MPLS networks[1].

A multipoint connection(MC) is a virtual topology, defined by routing table entries and other state information within the network, that supports the delivery of multicast data among multiple computers. The topology of an MC is usually a tree, whose edges represent communication links. Routing table entries, maintained at switching elements in the network, are used to forward traffic destined for the group along branches of the tree. A multicast protocol defines how the MC is established and maintained. A number of multicast protocols have been proposed for use in the Internet, including the DVMRP, MOSPF, CBT, and PIM.

In an important class of multicast protocols, called core-based forwarding (CBF) protocols, a core node is associated with each multicast group. The topology of the multicast tree, shared by the group, is defined to be the union of the member-to-core shortest paths. Messages destined for the group are first routed to the core node, from which they are distributed along tree branches to group members. Prominent CBF multicast protocols include the Core-based Tree (CBT)and the Protocol Independent Multicast-Sparse Mode (PIM-SM).

In multicast routing,the incorporation of a QoS guarantee is an important issue, together with scalability and reliability. In general,existing core-based multicast routing protocols construct only the shortest paths between the core and members in a multicast group without taking into consideration the QoS. If there is little variation in the QoS for different cores, random core selection is sufficient. If, however, the variation is substantial, more sophisticated core selection methods are required. Although various core selection algorithms have been proposed, there are few algorithms that take the QoS constraints into consideration[2].

Core-based routing provides a scalable multicast delivery to multi-sender multicast applications since only one multicast data delivery tree rooted at a single core is constructed per group regardless of the number of senders. Data destined to the multicast group is routed towards the single core. The core, then, distributes the data to all the group members via the multicast tree. Existing core-based routing with QoS support has two major drawbacks. First, it may be difficult for users who are group members to specify exact values of the desired service quality as in the existing work. Second, routing with a single core may not satisfy QoS requirements of many distributed group members. As a result, the service can be seriously affected. For instance, a video conference may not be worthwhile if important group members cannot participate with their desired QoS requirements.

Table 1: Example of end-to-end delay classes

Delay class	Upper-bound delay (ms)
very short	66
short	83
medium	101
best effort	∞

To alleviate the first drawback, Putthividhya[3,4,5,6] introduce a novel application level service class framework. In this framework, a multicast group is associated with a set of pre-defined service classes. A group member selects one of these service classes to indicate the desired service quality. Each of the classes specifies a different bound of the same end-to-end QoS metric such as delays or transmission bandwidth. The choices of the bounds for the service classes and the number of service classes depend on the types of applications. Table 1 shows possible end-to-end delay classes for virtual collaboration applications. Putthividhya's application-level service classes differ from network-level service classes in Differentiated Services and the reduced service-set architecture. A network-level service class is transparent from users. In the framework, a multicast application offers a user a set of service classes, making it easier to select his/her desired service quality. To address the second drawback, Putthividhya investigate the use of as many cores per group as necessary in QoS core-based routing under the service class framework. Since core selection is the first and necessary step for core-based routing.Putthividhya formulate a new QoS core selection problem to select the smallest set of cores for a multicast group that maximizes the number of group members with guaranteed end-to-end QoS requirements.

In this paper,we propose a core selection algorithm to the core selection problem in Core-based routing with QOS requirements.we select the smallest set of cores from the set of candidate cores such that the number of group members with satisfied end-to-end QoS requirements is maximized.Hence, our work is different from existing core selection algorithms that choose only a single core per group [1,2,7] and those that use multiple cores per group without QoS support [8,9].

The remainder of the paper is organized as follows. In Section 2, we propose core selection problem in core-based routing with the QOS requirements. In Sections 3, we discuss our algorithms performances. Finally, we conclude our work in Section 4.

2 Core Selection with QoS Support

We first formulate a new QoS primary core selection problem and discuss the solution to the problem. Next, we present our proposed core selection algorithm.

2.1 Service Class Framework and QoS Core Selection Problem

We revisit Table 1 showing four possible end-to-end delay classes for virtual collaboration applications. Each class in the table specifies the upper bound of the end-to-end delay guaranteed to the members requesting for that class. Except for the best-effort class, the bounds specified in the rest of the service classes are acceptable to users of virtual collaboration applications. For example, the users requesting for the *very short* delay class expect to experience the end- to-end delays of at most 66 ms. On the other hand, the requesting users of the short delay class anticipate to experience at most 83 ms end-to-end delay. Users who do not require a high degree of interactivity may choose the class with a longer guaranteed end-to-end delay given a cheaper service fee as an incentive. Users are either guaranteed the bounds of the requested service classes or denied the service if re-negotiation for another service class is not supported[3,4,5,6].

We define the following terms to be used throughout the paper. A *member router* and a *sender router* are defined as a multicast-capable router designated by a group member and a sender, respectively. One router acts as both types if designated by both senders and group members. A member router subscribes to all the service classes requesting by its designating group members. However, senders are not required to join the group. *A member router j covers a member router i* subscribing to the service class c if the bound corresponding to the class can be guaranteed when the member router i receives multicast data from all the sender routers through the router j. In other words, the router j is capable of being a core for the router i for this service class. The set of member routers that subscribes to the service class c and is covered by the member router j is called the *class c covering set of j* and denoted as $cover_c(j)$. We focus on guaranteeing end-to-end QoS requirements along the paths between sender routers and member routers. We assume that the QoS requirements between a group member and its designated router are always assured since they are typically on the same LAN.

Employing as many cores as necessary has two additional advantages. First, they can be backups for each other should some cores fail. Second, when many routers simultaneously join the group, they can be directed to different existing cores to prevent any core from becoming a hot spot. Nevertheless, the advantages of using as many cores per group as necessary do not come for free. The sender-to-core traffic increases proportionally to the number of cores since each sender router has one unicast stream to each core. To minimize the cost, we minimize the number of cores while maximizing the number of group members with assured QoS requirements.

This work addresses the core selection problem, which is to find the network node (i.e., a router) whose use as the core of the multicast group results in the better multicast tree, with respect to one or more performance metrics. Performance metrics of interest include network resource usage, end-to-end delay for multicast packets, the time required for new members to join the group, and network congestion. The core selection problem is related to the well-known Steiner tree problem, in which a tree of minimum cost is sought to connect a subset of vertices in a graph. In the core selection problem, however, the multicast tree topology for a given group must be the union of member-to-core shortest paths. As such, the topology of the tree is completely determined by the selection of the core node and the distance metric used in the underlying network routing protocol[1,2,7].

Let SC be the set of pre-determined service classes offered to a multicast group. Let R be the set of all member routers of the multicast group. In our study, all the member routers are eligible candidate cores of the multicast group.

Problem Statement: *Given the service class framework and* $cover_c(j) \forall C \in SC, \forall j \in R$*, select the smallest set of cores from the set of candidate cores such that the number of group members with satisfied end-to-end QoS requirements is maximized[3,4,5,6].*

The problem statement cannot be formulated as an integer programming problem since a member router may subscribe to more than one service classes. A binary decision variable in an integer programming problem is inadequate to identify which of the service classes a candidate core covers the member router.

The problem statement and our solution can be generalized to work in an environment in which pre-defined service classes are not available. In this case, we treat the distinct QoS values specified by group members as the different bounds of various service classes. The number of service classes is equal to the number of the distinct QoS values. Hence, our solution is applicable without any modification.

The set-covering problem [10] is polynomial-time reducible to our QoS core selection problem. Hence, our problem is as hard as the set covering problem known to be NP-hard and is also NP-hard. Due to limited space, the proof is shown in Reference [3].we propose a core selection algorithm to core selection problem in Core-based routing with QOS requirements.we select the smallest set of cores from the set of candidate cores such that the number of group members with satisfied end-to-end QoS requirements is maximized.

2.2 QoS Core Selection Protocol

We present our core selection protocol for the core selection prblem discussed in Sections 2.2.

Step 1: Registration: Each member router submits to a pre-determined bootstrap router the number of its designating group members in each service class. The sender routers also register to the bootstrap router. The bootstrap router unicasts the information received from all the member routers to every sender router.

Step 2: Distributed Resource Reservation: Each sender router independently finds end-to-end QoS guaranteed paths from itself to all the member routers via each candidate core for every service class. This is per-formed using a modified QoS unicast routing protocol that reserves resources along the paths.To prevent any single candidate core from becoming a hot spot, each sender router works with the candidate cores in a random manner.

Step 3: Core Selection: Each sender router unicasts to the bootstrap router the information about end-to-end QoS-guaranteed paths which have been successfully constructed in Step 2. The bootstrap router determines the final set of cores from the received path information as follows.

Step 3(a): Derive all the covering sets $cover_c(j)$, $\forall j \in R$ and $\forall c \in SC$. A member router m_i is covered by a candidate core $m_j(m_i \in cover_c(j))$ if a QoS-guaranteed path from every sender router to m_i via m_j has been successfully constructed in Step 2.

Step 3(b): We use a iterations according to practical situation no consider the service class.

In ith iteration,at first the candidate core with the ith largest group members is selected.Then if there is a tie, the candidate core with the largest group members is selected. If there is still a tie, the candidate core with the lowest ID derived from its IP address is chosen.

In the last section,we contract our algorithm with Putthividhya's algorithm and simulations result show our algorithm better than the Putthividhya's.

Step 3(c): Invoke the core merging algorithm that determines the final set of cores in two steps.

- Core Union: Union the set of cores of all the service classes.

• Core Set Reduction: The selected core m_i can be further removed from the core set after the union if there exists another core m_j that can cover all the member routers in every class c covering set of m_i,$c \in SC$. We also ensure that each member router is assigned to only one core in this step.

Step 3 (d): The bootstrap router informs (i) all the sender routers about the final core set of the group and (ii) the selected cores about the member routers assigned to them and the corresponding path information.

After receiving the end-to-end path information from the bootstrap router, each core determines the best core-to-end path to each of its assigned member routers. A multicast tree rooted at the core is constructed using our new multicast tree construction algorithm.

3 Performance Study

In the section,at first we give a example which have compared our algorithm with CCSA[3,4,5,6],then the simulations show that our algorithm is better than CCSA.

3.1 Examples comparison with CCSA

First,we give a example show that the selected cores in no considering delay class situation is less than in considering delay class situation. Figure1(a),The A,B,C,D,E are the cores,and the number 1,2 refer the router delay class.class 1 is tighter than class 2.Using CCSA algorithm,first consider tighter delay class,so first select core B(or D) since its class 1 covering set is the largest.Then the core D(or B) since its class 1 covering set is second.Now consider class 2,select core A and E in same method.So CCSA algorithm needs 4 cores(B,D,A,E).In our algorithm,no consider delay class,so first select core A(or E) since covering set is the largest.Then select E(or A),at last select core C.Our algorithm need 3 core(A,E,C) fewer than CCSA.

Second,we give a example show that the selected cores in using the iterations situation is less than in no using the iterations situation. Figure1(b),A,B,C are the cores,and suppose the routers(blank circle) are in same delay class,so in CCSA algorithm first select core B since its covering set is the largest.Then core A and C.The CCSA algorithm need 3 core(B,A,C).When use our algorithm,first iteration is the same as CCSA algorithm.In second iteration,first select core A since its covering is second,then select core C.Second iteration need 2 cores(A,C) fewer than CCSA.

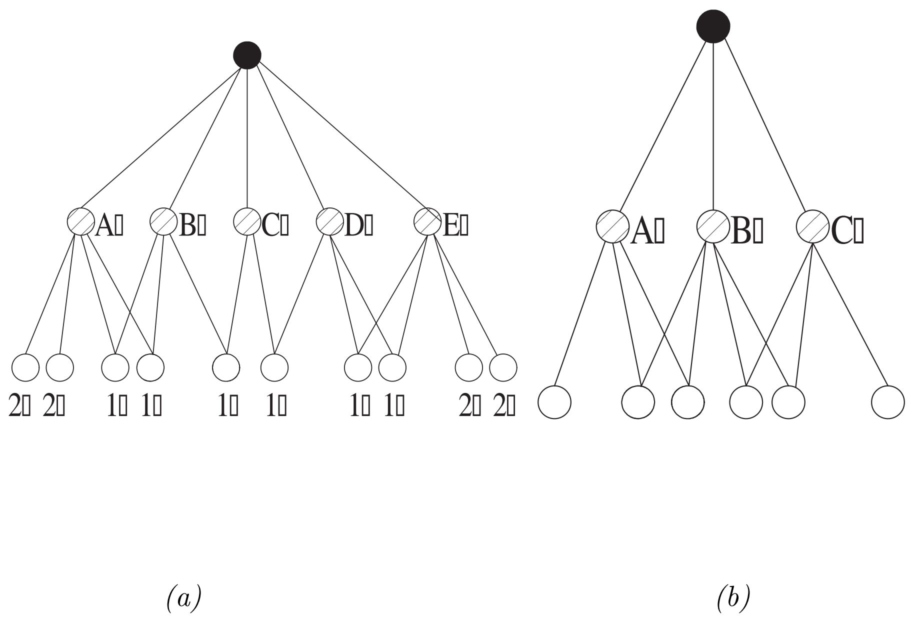
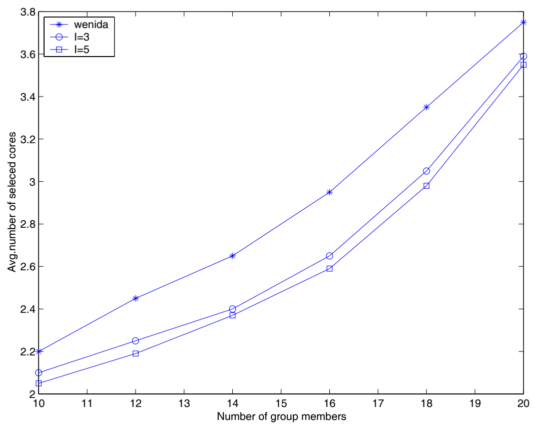


Fig 1.Examples comparison with CCSA

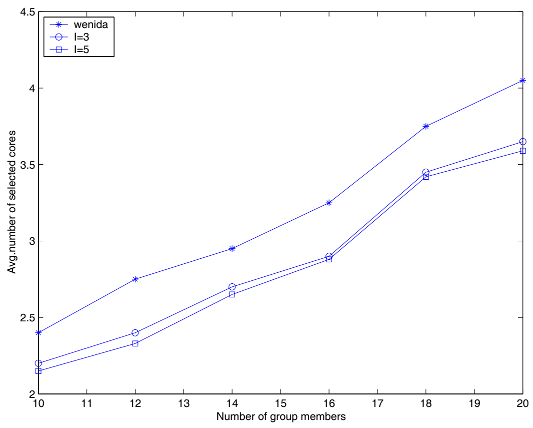
3.2 Performance comparison with CCSA

The our algorithms described in this paper has been tested on several randomly generated networks based on the Waxman's algorithm[11].In the algorithm, n nodes are randomly distributed over a rectangular coordinate grid.Each node is placed at a location with integer coordinates. The Euclidean metric is then used to determine the distance between each pair of nodes.On the other hand,edges are introduced between pairs of nodes u,v with a probability that depend on the distance between them. The edge probability is given by $P(u, v) = \beta exp(-d(u, v)/\alpha L)$,where $d(u, v)$ is the distance from node u to v,L is the maximum distance between two nodes,and α and β are parameters in the range (0,1).Larger values of β result in graphs with higher edge densities,while small values of α increase the density of short edges relative to longer ones.

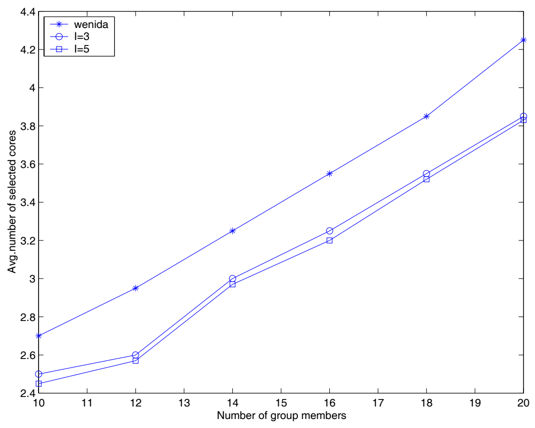
We investigate the performance of the our proposed heuristic algorithm through computer simulations.Waxman's method is employed to generate four types networks,i.e,50-,75-,100-,125-nodes,each of which includes 10 instances.



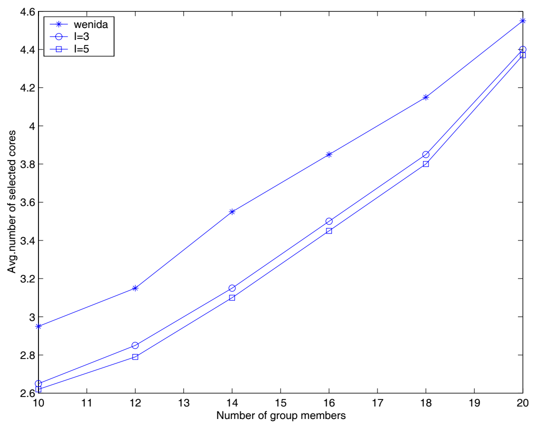
(a)Network siza=50



(b)Network siza=75



(c)Network siza=100



(d)Network siza=125

Fig 2.Performance comparison with CCSA

In Fig 2,we have compared our algorithm with CCSA(Putthividhya's algorithm),and our algorithm have three,five iterations in the performance, respectively. From the figure we are easy found that our algorithms's Average numbers of selected cores are less than CCSA,and our algorithms's Average numbers of selected cores with five iterations is less than with three iterations.This is show that our algorithms are better than CCSA,and our algorithm with five iterations is better than our algorithm with three iterations. In general,our algorithms have less selected cores performance and are feasible and effective.

4 Conclusion

For core selection problem in Core-based routing with QOS requirements, we propose a core selection algorithm.We select the smallest set of cores from the set of candidate cores such that the number of group members with satisfied end-to-end QoS requirements is maximized. Simulation results show our algorithms has features of well performance of less selected cores,and are feasible and effective.

References

[1] E.Fleury,Y.Huang,On the Performance and Feasibility of Multicast Core Selection Heuristics,Networks,2000,35(2),145-156.

[2] S.M.Chung,C.H.Youn,Core selection algorithm for multicast routing under multiple QoS constraints,Electronics Letters,2000,36(4),378-379.

[3] W.Putthividhya,W.Tavanapong,M.Tran,J.Wong. A novel core selection with end-to-end QoS support for multi-sender multimedia multicast applications. Technical Report: Department of Computer Science, Iowa State University, (TR-03–07), 2003.

[4] W.Putthividhya,W.Tavanapong,M.Tran,J.Wong, Distributed Core Selection with QoS Support, IEEE International Conference on Communications 2004 (ICC'04),June 2004.

[5] W.Putthividhya,W.Tavanapong,M.Tran,J.Wong,Core selection with end-to-end QoS support,Proceedings of the 2004 ACM symposium on Applied computing.

[6] W.Putthividhya,W.Tavanapong,S.K.Wong,Core-based Routing with QoS Support for Distributed Interactive Multimedia Applications,IJCSNS International Journal of Computer Science and Network Security, 6(1), January 2006.

[7] D.Thaler,C.Ravishankar. Distributed center-location algorithms. IEEE journal on Selected Areas in Communications, 15(3):273-276, April 1997.

[8] L.Blazevic,J.L.Boudec. Distributed core multicast (DCM): a multicast routing protocol for many groups with few receivers. ACM SIGCOMM Computer Communication Review, 29(5):6-21, October 1999.

[9] D.Zappala,A.Fabbri,V.Lo. An evaluation of shared multicast trees with multiple cores. Journal of Telecommunication Systems Kluwer Academic Publishers, 19(3-4), March 2002.

[10] T.Cormen,C.Leiserson,R.L.Rivest. Introduction to Algorithms. McGraw Hill, 2000.

[11] B.M.Waxman,Routing of multipoint connections,IEEE Journal on Selected Areas in Communication 6(9)(1988) 1617-1622.

[12] A.Karaman,H.Hassanein.QoS-constrained core selection for group communication. Computer Communications,30(2007),1-13.

[13] Y.Huang, E.Fleury,P.K.McKinley.Lcm: A multicast core management protocol for link-state routing networks. IEEE International Conference on Communications, 2:1197C1201, June 1998.

[14] A.Ballardie,Core Based Trees (CBT version 2) Multicast Routing C 893 Protocol Specification, RFC 2198, September 1997.

[15] A.Ballardie,Core Based Trees (CBT) Multicast Routing Architec- 895 ture, RFC 2201, September 1997.

[16] H.Wang,et al.TSDLMRA:an efficient multicast routing algorithm based on Tabu search.Journal of Network and Computer Applications 27(2) (2004) 77-90.

[17] H.F.Salama,et al.Evaluation of multicast routing algorithms for real-time communication on high-speed networks. IEEE J Sel Areas Commun 15(3)(1997) 332-345.

[18] Z.wang,J.Crowcroft.Quality of service for supporting multimedia applications. IEEE JSAC, 14 (1996) 1228-1234.

Received: July 21, 2007