DQM: An Overlay Scheme for Quality of Service Differentiation in Source Specific Multicast

Ning Wang, George Pavlou
Centre for Communication Systems Research, University of Surrey
Guildford, United Kingdom
{N.Wang, G.Pavlou}@eim.surrey.ac.uk

Abstract. In this paper we propose a new scheme named *DQM* (Differentiated *QoS* Multicast) based on the Source Specific Multicast (*SSM*) [7] model in order to provide limited and qualitative *QoS* channels to support heterogeneous end users. Similar to the DiffServ paradigm, in *DQM* the network is configured to provide finite *QoS* service levels to both content provider and receivers. Based on the Service Level Agreement, both the content provider and group members should select a specific *QoS* channel available from the network for data transmission, and in this case arbitrarily quantitative *QoS* states are eliminated. Moreover, we use the group address *G* contained in the (*S, G*) tuple in *SSM* service model to encode *QoS* channels, and data packets that belong to the same *QoS* channel identified by a common class *D* address can be treated aggregately, and this can be regarded as an overlay solution to Differentiated Services, specifically for source specific multicast applications.

1 Introduction

In contrast to the current Internet, applications with Quality-of-Service (*QoS*) requirements will be an important aspect in the next generation of the Internet. In addition, applications based on group communication will also become widespread. Given this expected evolution, the situation in which the Internet is uniquely dominated by point-to-point applications based on the Best Effort (*BE*) service level will change in the future.

Multicasting is an efficient approach for group communications, and the recently proposed Source Specific Multicast (*SSM*, [7]) model has been considered to be a promising solution for the development of one-to-many applications in a large scale. In *SSM* each group is identified by an address tuple (*S*, *G*) where *S* is the unique *IP* address of the information source and *G* is the destination channel address. Direct join requests from individual subscribers create a unique multicast tree rooted at the well-known information source, i.e., *SSM* defines (*S, G*) channels on per-source basis. In this model, the scalability problems of *IP* multicast such as address allocation and inter-domain source discovery are not deployment obstacles any more. Due to its simplicity and scalability, *SSM* is expected to see significant deployment on the Internet in the near future, especially for single source applications.

On the other hand, the provisioning of *QoS* requirements in a scalable manner is another major research dimension towards the next generation of the Internet. An efficient solution is to classify traffic into finite service levels and treat packets that belong to the same *QoS* service level in an aggregate manner. Differentiated Services (*DiffServ*) [1] is a typical example of this approach and is considered a scalable scheme for deploying *QoS* widely. Research efforts have also addressed the problem of applications with heterogeneous *QoS* requirements given the potentially different capacity of individual receivers. D. Yang *et al* proposed Multicast with *QoS* (*MQ*) [12] as an integrated mechanism with the consideration of *QoS* routing, resource reservation and user heterogeneity. This genuine receiver-initiated approach inherits some basic characteristics of *RSVP* [3], such as quantitative *QoS* guarantee and resource reservation merging from heterogeneous end users. It should be noted that *MQ* also requires that on-tree routers maintain state on a per-flow basis for end-to-end *QoS* guarantees, and this aspect still leaves the scalability issue problematic.

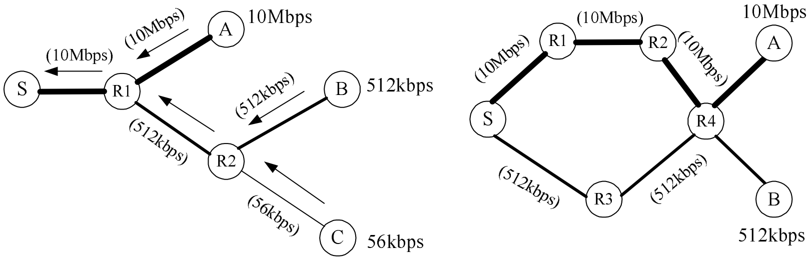
In this paper, we propose a new framework called Differentiated *QoS* Multicast (*DQM*) to support *qualitative* service levels (e.g., Olympic Services) based on the Source Specific Multicast model. The basic characteristic of *DQM* is as follows: First, qualitative *QoS* states are directly encoded in the class D address and is centrally managed by the *ISP*, so that core routers inside the network remain stateless regarding *QoS* service classes. Second, differentiated level of *QoS* demands for the specific information source is merged in the distribution tree, and data packets from different sources but belonged to the same *QoS* service level identified by a common multicast group address can be treated aggregately. Moreover, a pre-defined number of classes of services by the *ISP* make it easier to provision network resources for each *QoS* aggregate, and this is in the same flavor of the classical Differentiated Services. From this point of view, the proposed *DQM* model can be regarded as an overlay solution of DiffServ, specifically for source specific multicast applications.

2 Supporting Applications with *QoS* Heterogeneity

2.1 The *MQ* Approach

Being an integrated solution, *MQ* sets up a multicast distribution tree with quantitative *QoS* requirements, and makes explicit bandwidth reservation for each group member during the phase of tree construction. When there exist heterogeneous receivers, resources are reserved up to the point where the paths to different receivers diverge. When a join request propagates upstream towards the source, it stops at the point where there is already an existing *QoS* reservation that is equal to or greater than that being requested. Fig. 1 basically illustrates how different resource reservations are merged along the multicast join procedure. Suppose the requests from receiver *A*, *B* and *C* demands 10Mbps, 512kbps and 56kbps bandwidth respectively, their reservations are merged to the highest request at each hop as shown in the figure. *MQ* can also adapt to resource consumption with dynamic group membership. For example, if an on-tree router detects that the departing receiver originally requested the highest *QoS*, it will automatically shrink its reservation or even reshape the distribution tree to exactly satisfy the remaining participants. In Fig. 1(b), we can find that when receiver *A* with the bandwidth requirement of 10Mbps wants to leave the multicast session, the remaining receiver *B* with 512kbps requirement will switch from the original "shared" path (S→R1→R2→R4) with the capacity of 10Mbps to a shorter one (S→R3→R4) which still satisfies its *QoS* demand for bandwidth optimization purpose.

On the other hand, the mechanism for network resource allocation is in an accumulative fashion, i.e., bandwidth is reserved in sequence for various incoming *QoS* requests until the link becomes saturated. This approach is simple but might not be efficient in bandwidth allocation especially in case of highly dynamic group membership. From the deployment point of view, each on-tree router needs to maintain not only group states but also the quantitative *QoS* demands for its downstream receivers, and this imposes heavy overhead in a similar fashion to *RSVP*.

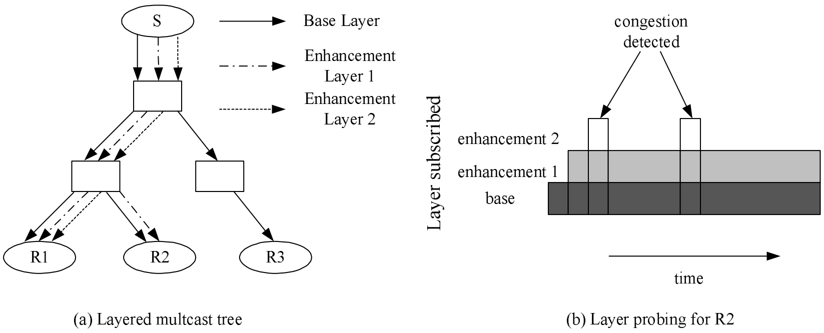


(a) (b)

Fig. 1 *MQ* group join and tree reshaping

2.2 Layered Transmission

This approach is particularly useful for Internet *TV* applications since it relies on the ability of many video compression technologies to divide their output stream into layers: a *base layer* as well as one or more *enhancement layers*. The base layer is independently decoded and it provides a basic level of quality. The enhancement layers can only be decoded together with the base layer to improve the video quality. The source can send individual layers to different multicast groups and a receiver can join the group associated with the base layer and as many layers for enhancement as its capability allows. Receiver-Driven Layered Multicast (*RLM*) [8] is a typical example for layered video transmission. Fig. 2 illustrates the working scenario of *RLM*, and Fig. 2(b) describes how receiver *R2* "probes" to subscribe to additional enhanced layers.

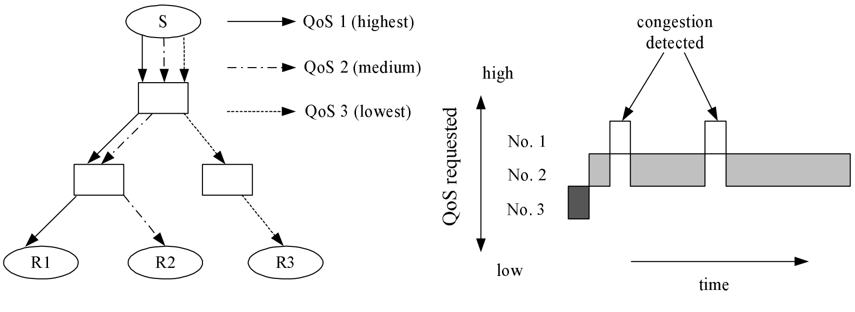


(a) Layered multcast tree (b) Layer probing for R2

Fig. 2 Layered transmission (*RLM*)

2.3 Replicated Transmission

It should be noted that not all types of multimedia streams can be encoded into layers as described above. An alternative approach is replicated transmission that is applicable to generalized type of multimedia applications. In this approach, the information source keeps a finite number of streams carrying the same content but each targeted at receivers with different capabilities. In a similar fashion to layered transmission, the data source assigns different multicast groups to each of the maintained streams, and receivers may move among them by subscribing to the corresponding group address. A typical example of replicated transmission is Destination Set Grouping (*DSG*) [4]. Fig. 3 describes how *DSG* works in a heterogeneous environment.



(a) Replicated multcast tree (b) QoS probing for R2

Fig. 3 Replicated transmission (*DSG*)

3 Basic *DQM* Framework

From the previous section, we can find that *MQ* provides a type of arbitrary bandwidth guaranteed service, while *DSG* and *RLM* adopts differentiated services to heterogeneous end users. From the viewpoint of scalability, the latter two approaches incur lighter group state overhead inside the network. On the other hand, in both *RLM* and *DSG* it is external sources that decide individual *QoS* classes, and thus it is difficult to perform aggregated data treatment for multiple sources inside the network since the *QoS* definition and configuration of each source/group session is different.

In this paper we propose a new transmission scheme, called Differentiated *QoS* Multicast (*DQM*), which can be regarded as the integration of Source Specific Multicast (*SSM*) and the Olympic Service model in DiffServ. From a viewpoint of the *ISP*, it provides external source/receivers with a finite class of services (e.g., gold service, silver service and bronze service), each of which is uniquely encoded into a class *D* address (within subset of 232/8) in the *SSM* model. In such a situation, the interpretation of *SSM* address tuple (*S, G*) becomes straightforward: *S* identifies the address of the information source and *G* stands for the *QoS* service level (we name it *QoS* channel) that is available from *S*. Once receivers have decided the source address *S* and the class address *G*, they will directly send (*S, G*) join requests to the source.

Since the provided service levels are centrally managed by the *ISP* instead of individual sources, this type of mapping between *QoS* class and group address does not introduce any scalability problem when the number of external sources increases (i.e., total number of *QoS* channels is independent of the number of external sources). The Bandwidth Broker (*BB*) in the network is responsible for computing available bandwidth resource and deciding whether or not to accept new data injections from sources as well as join requests from group members. When a new subscriber wants to receive data from the source *S*, it will first negotiate with *BB* on *QoS* channel selection. If the *BB* is able to allocate sufficient bandwidth to graft the subscriber to the existing source specific tree in the requested channel, the (*S, G*) join request is accepted. If the *BB* cannot allocate the required bandwidth for including the new subscriber into the *QoS* channel, the join request will be rejected, and the subscribers may subscribe to other *QoS* channels by sending different (*S, G'*) join requests. The relationship between sources/subscribers and the *ISP* network is described in Fig. 4.

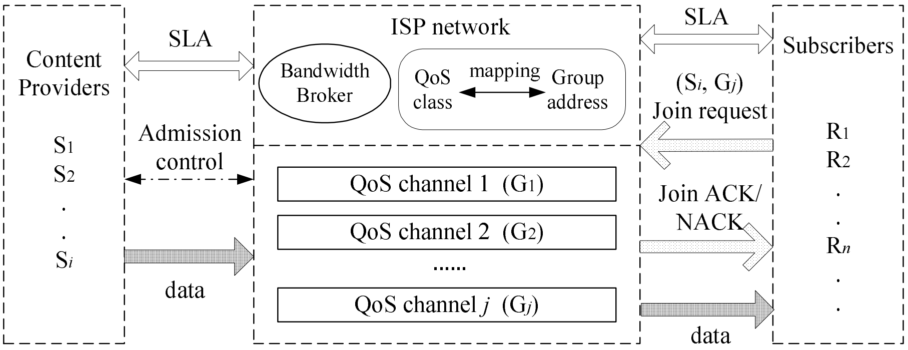


Fig. 4 *DQM* Framework

When the join request has been approved, the new subscriber will combine the *IP* address of *S* and group address *G* identifying the desired *QoS* channel into a (*S, G*) tuple, and send this join request towards the source *S*. If this (*S, G*) join reaches an on-tree router with group state (*S, G'*) where *G'* identifies a higher *QoS* channel, then this join request is terminated at this router. In this case, a new flow is branched from the existing source specific tree and leads to the new subscriber. A sender simply needs to inject a single data stream into the network with the highest *QoS* channel requested. Fig. 5 presents the basic description of the proposed *QoS* merging in *SSM* service model. It is worth mentioning that this type of *QoS* merging is significantly different from both *MQ* and *RSVP* explicit reservation. *MQ* and *RSVP* target to achieve end-to-end *QoS* guarantee with quantitative reservation style and hence the relevant state needs to be maintained for each flow with various *QoS* demands. In the scenario of differentiated *QoS* classes, not only group members have to select one of the available channels provided by the *ISP*, but also information sources should conform to the *ISP's* channel configuration profile. In this case group states scalability inside the network is achieved since external sources are not allowed to provide their own *QoS* channel configuration as they do in *DSG* and *RLM*. Also note that data treatment inside the network is decided by the class *D* address *G*, i.e., within each router data packets from different sources *S* but with common class *D* address *G* can be treated aggregately. In this sense, the group address *G* takes additional role similar to that of the Differentiated Services Code Point (*DSCP*) in DiffServ networks.

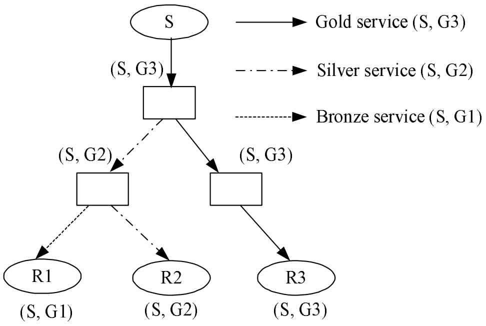


Fig. 5 *DQM* group state maintenance

The advantages of the proposed *DQM* scheme are as follows. Most importantly, it solves the fundamental confliction between stateless DiffServ service model and state based IP multicast semantics. In DiffServ model, core routers do not maintain any *QoS* states for individual applications/flows, and data treatment is indicated inside each packet header. On the other hand, the basic mechanism of IP multicast is to keep group states inside the network so as to enroute data to active group members. In *DQM*, *QoS* states are directly encoded into multicast address and maintained inside the network, as originally needed in IP multicast, and hence no other *QoS* information need to be kept at core routers. Otherwise, if we don't use group address to identify *QoS* channels in the join request, this information should be contained elsewhere, for instance, in the option field of the join request packet. When this join request is heading for the information source, each of the routers it has passed through should record the *QoS* requirements information and associate it with the downstream interface from which the join request has been received. This is necessary because otherwise *DSCP* contained in the data packet cannot be modified when the packet reaches the branching point of the source specific tree where heterogeneous *QoS* classes meet each other. By recording the *QoS* class information at core routers, when the group data comes from the upstream interface, the on-tree router exactly knows through which *QoS* class it should forward the packets on its different downstream interfaces. No doubt, this approach requires that core routers maintain *QoS* information at its downstream interfaces in addition to the plain (*S, G*) state, but this does not conform to the basic *QoS* stateless requirement of DiffServ service model. Second, service differentiation is centrally defined and managed by the *ISP* instead of individual sources, as it is done in *DSG* and *RLM*, traffic from different sources but with identical *QoS* class can be treated in an aggregated fashion inside the network. Finally, in contrast with the "come and use" strategy of bandwidth allocation in *MQ*, *DQM* allows an *ISP* to allocate network resources specifically to individual *QoS* channels according to the forecasted traffic demands from each of them, so that the traffic loading can be improved by the more flexible bandwidth configuration. However, there is a restriction regarding this approach. Since the *QoS* channel is source specific, it is impossible for a single source with a unique *IP* address *S* to send multiple data streams with different contents. In the classic *SSM* model, an information source can be simultaneously involved in multiple groups because (*S, G1*) and (*S, G2*) are completely independent with each other. A short-term solution to this restriction is to use different unicast IP source address for each group session.

4 *DQM QoS* Channel Maintenance

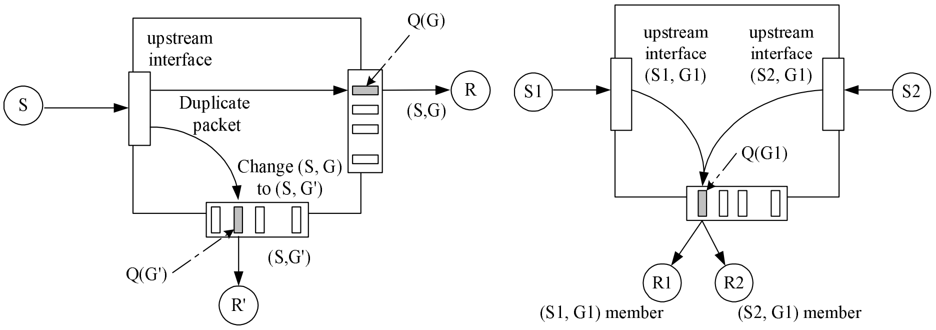
4.1 Data Forwarding

Current router implementation for service differentiation adopts priority queuing technologies such as Class Based Queue (*CBQ*) and Weighted Faired Queue (*WFQ*). For example, in DiffServ networks data packets marked with different *DSCP* value are treated in queues with different priority for scheduling. Similarly, in *DQM* core network bandwidth is divided specifically for each *QoS* channel and data packets from different channels (distinguished by class D address) are scheduled in the corresponding priority queues. In this section we basically describe the working mechanism of routers that can support the relevant functionality of *QoS* channel differentiation.

Once a router receives (*S, G*) join requests with different values of *G* that are associated with various *QoS* channels from subscribers, it will merge all of them and only send a single (*S*, G_m) join request towards *S*, where G_m is the class *D* address associated with the highest *QoS* channel being requested. Here we define the interface from which a join request is received as the *downstream interface* and the one used to deliver unicast data to the source as the *upstream interface*. When the router receives group data from its upstream interface it will take the following actions to forward the packets (also shown in Fig. 6(a), suppose *QoS*(*G*)>*QoS*(*G'*)):

(1) Look up the group state(s) associated with the source *S* on each downstream interface and duplicate the packet where necessary.
(2) Copy the value of *G* contained in the (*S, G*) state at each downstream interface to the *IP* destination field in the duplicated packet (if the two are not consistent).
(3) Assign the data packet to the priority queue associated with relevant *QoS* channel at the downstream interface based on the (*S, G*) channel state.

Step (2) is necessary since the value of *G* contained in the packet indicates how this packet will be treated in the next hop of on-tree router. Remember that the group states are created by (*S, G*) join requests for different *QoS* channels, and the way data packets are treated at each router is uniquely identified by the value of *G* contained in the (*S, G*) state, and in this way data can be forwarded according to the *QoS* requests from individual users. On the other hand, data packets from different sources *S* but with the same class *D* address in their (*S, G*) address tuples are treated aggregately in the corresponding queues. To achieve this, the *ISP* should also make the configuration such that each priority queue is associated with a group address on downstream interfaces (shown in Fig. 6(b)). This figure also illustrates how data from different sources but with a common group address is treated aggregatively in a specific queue of a core router.



(a) Data duplication (b) Data aggregation

Fig. 6 *DQM* Forwarding behaviour at core routers

4.2 Dynamic Group Membership

In most multicast applications, subscribers join and leave frequently throughout a session. In this section we will discuss on how *QoS* channel merging for a common source is performed with dynamic group membership. On the other hand, individual priority queue should be implemented with proper bandwidth allocation for each *QoS* channel, thus the joining path of same source-destination pair might not be exactly the same for all (S, G) channels. Path computation should also consider the specific bandwidth availability of the subscribed *QoS* channel, and we name this Per Channel *QoS* routing in *DQM*.

(1) *QoS* channel subscription

When a (S, G_i) join request reaches a router that (i) has already received traffic from the source S with the same or a higher *QoS* channel, i.e., with group state of (S, G_j) where $G_i \leq G_j$[1] and (ii) the corresponding priority queue at the interface from which the join request is received has sufficient bandwidth, then the join procedure terminates and this interface obtains group state (S, G_i). Thereafter, data packets from S are duplicated and forwarded to this interface with the class D address of the new packets modified from G_j to G_i. In this way, a new tree branch is grafted to the current *QoS* channel that has equal or higher service level.

If the (S, G_i) join request reaches a router with the highest available *QoS* channel (S, G_j) where $G_i > G_j$ (i.e., a router with lower *QoS* channel for S), the join will continue to explore a new path that satisfies the requirement of the (S, G_i) channel subscription. Once a path with desired *QoS* channel has been set up and this particular router has received the traffic from (S, G_i) channel, it will tear down the (S, G_j) channel on the original path with lower *QoS* level. It should also be noted that the

---

[1] In this paper we assume that higher class D address is associated with higher *QoS* channel, i.e., $G_i > G_j \leftrightarrow QoS(G_i) > QoS(G_j)$

procedure of tearing down (S, G_j) channel might invoke another join request from an on-tree router where (S, G_j) is the highest local channel it maintains and there exist other channels with lower *QoS* channel.

In Fig. 7, we assume that initially there already exists a single *QoS* channel constructed by (*S, G2*) subscriptions from both receivers *R1* and *R2* (Fig.7-a). After some time router *D* receives a (*S, G1*) subscription from *R3* where $G1<G2$, i.e., a subscription with lower *QoS* channel is received. In this case *D* will send a join request towards *S* and this join request will terminate at router *B* that has already received group data from *S* in a higher channel (shown in Fig. 7-b). In Fig. 7-c, we assume that router *E* receives a (*S, G3*) join request from *R4* where $G3>G2$. In this case a new path with a higher *QoS* channel is constructed as shown with the solid line in the figure. When router *E* receives data traffic from *S* in (*S, G3*) channel, it will tear down the original (*S, G2*) channel back to *S*. When router *B* has detected the pruning, it finds that it has also maintained a lower *QoS* channel for *R3*, namely (*S, G1*). Therefore, router *B* will first send a (*S, G1*) join request back to *S*. When detecting that group data from *S* comes in the new channel (*S, G1*), router *B* will tear down the original (*S, G2*) channel on link *AB* as shown in Fig. (7-d). More detail on how *QoS* channels are torn down is presented below.

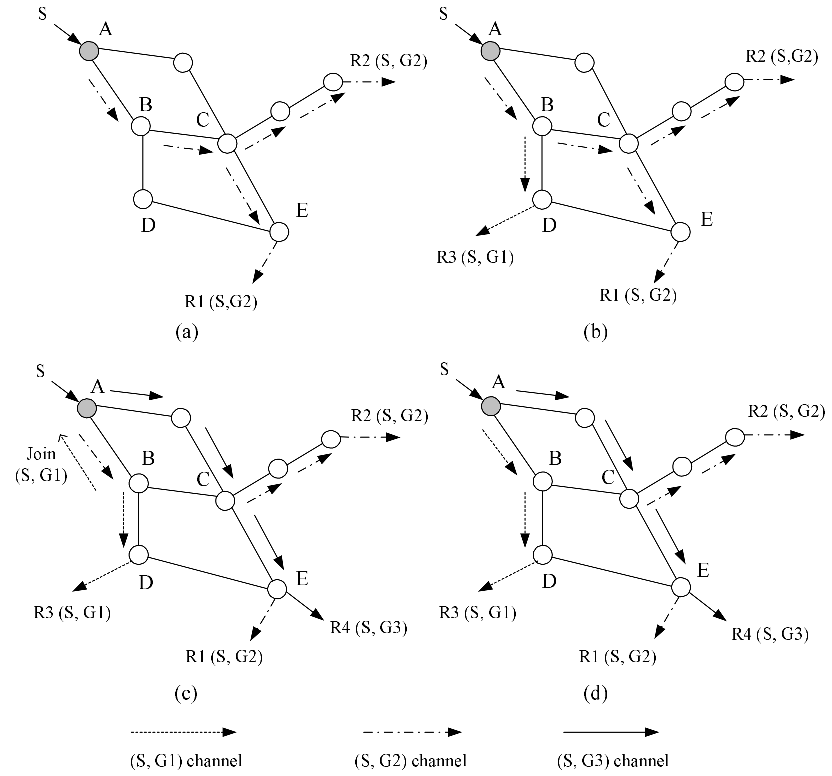


Fig. 7 Dynamic *QoS* channel subscription

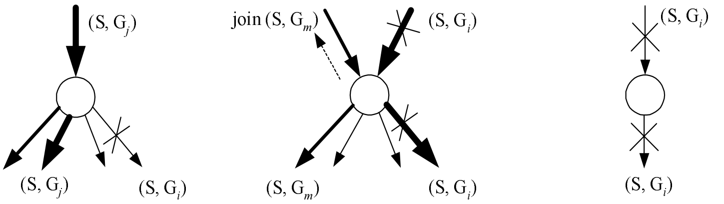
(2) *QoS* channel unsubscription

Suppose that a particular router is currently receiving traffic from source *S* with *QoS* channel (S, G_i). When it detects no (S, G_i) subscribers attached and wants to leave the channel, it will stop sending (S, G_i) join requests towards the source *S*. When the (S, G_i) state times out, the on-tree router will check all its downstream interfaces with *QoS* channels associated with *S*. There exist three possible cases as follows (illustrated in Fig. 8):

a. There exists at least one (S, G_j) state where $G_j \geq G_i$, the router simply stops forwarding traffic on (S, G_i) channel at the corresponding downstream interface, and it needs not to take any further pruning actions;

b. There does not exist any (S, G_j) state where $G_j > G_i$, the router will check the status of all the remained *QoS* channels associated with *S* and select class *D* address G_m that is mapped to the highest *QoS* channel being currently requested, and send (S, G_m) join request towards the source *S*. Once this router has received data traffic from the (S, G_m) channel, it will stop sending (S, G_i) join requests on its upstream interface;

c. If this is the last subscriber attached on the router, the router simply stop sending any (*S, G*) join requests towards the source and hence it breaks from the tree.



(a) (b) (c)

Fig. 8 Dynamic *QoS* channel unsubscription

Finally, it should be noted that, a boundary router issues join/leaving requests only when the first receiver for a new (*S, G*) session is joined or the last member leaves the group. This strategy of pushing group management to the edge of the network significantly reduces the frequency of reshaping delivery trees inside of the domain.

5 Simulation Results

5.1 Simulation Model

In this section, we evaluate the proposed scheme through simulation. We adopt the commonly used Waxman's random graph generation algorithm [11] that has been implemented in *GT-ITM*, for constructing our network models. This approach distributes the nodes randomly on the rectangular grid and nodes are 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 in the network. The parameters λ and ρ ranging (0, 1] can be modified to create the desired network model. A larger value of λ gives node with a high average degree, and a small value of ρ increase the density of shorter links relative to longer ones. In our simulation we set the values of λ and ρ to be 0.3 and 0.2 respectively, and generate a random network with 100 nodes with the source node being randomly selected.

In order to generate group dynamics, a sequence of events for *QoS* subscription/unsubscription are also created. A probability model is used to determine whether a request is for *QoS* subscription or unsubscription. The function

$$P_c=\frac{\alpha(N-m)}{\alpha(N-m)+(1-\alpha)m}$$

is defined for this purpose [11]. The function P_c is the probability that a *QoS* subscription is issued. In the function, m indicates the current number of subscribers while N identifies the network size. α ranging (0, 1) is the parameter that controls the density of the group (i.e., average number of subscribers). When a *QoS* subscription is issued, a node that is not in the multicast group is randomly selected for joining the session. Similarly a node is randomly removed from the current group when a *QoS* unsubscription request is triggered.

In our simulation, the average number of subscribers varies from 10 to 40 in steps of 5 (by setting the value of α). For simplicity we assume that there is at most one subscriber attached at each router. In addition, we also assume that the *ISP* provides three qualitative *QoS* channels, namely Gold Channel (*GC*), Silver Channel (*SC*) and Bronze Channel (*BC*), and that the subscription bandwidths for these three channels are 8Mbps, 4Mbps and 2Mbps per receiver respectively. Inside the network, the bandwidth capacity of each link varies from 10Mbps to 45Mbps in even distribution. The bandwidth capacity of each link is configured according to the following proportion: 50% for *GC*, 30% for *SC* and 20% for *BC* respectively. Among all the receivers, we suppose that 20% of them subscribe to *GC*, 30% to *SC* and 50% to *BC*. In our simulation we adopt *QOSPF* for receiver-initiated multicast routing that is introduced in [12] as the underlying routing protocol for each *QoS* channel. According to [12], *QOSPF* based multicast routing does not support user heterogeneity within a particular group, but in *DQM* such type of *QoS* heterogeneity is reflected by different (*S, G*) group identification. In this sense, *QOSPF* can still apply to per *QoS* channel routing in *DQM*, and different tree branches can be merged if the same source address *S* is discovered.

5.2 Performance Analysis

First of all, we investigate bandwidth conservation performance, and comparisons are made between *DQM* and that of building independent trees for each service level with disjoined *QoS* channels (e.g., *DSG* [4] in which the source maintains independent data streams for heterogeneous users simultaneously). Following that we focus on the capability of traffic engineering in terms of load balancing between *DQM* and *MQ* approaches. The simulation results regarding network utilization are also compared between *DQM* and *MQ* but overhead for group states maintenance is incomparable since the latter involves quantitative states for user heterogeneity.

In order to evaluate the network utilization, we define the bandwidth saving overhead for a particular channel *C* (*C* could be gold, silver and bronze service etc.) as follows:

$$O_C = 1 - \frac{U_{DQM}^{C}}{U_{DSG}^{C}}$$

where U_{DQoM}^{c} is the bandwidth utilization of channel *C* by *DQM*, and U_{DSG}^{c} is that by using non-hybrid tree schemes such as *DSG*. Similarly, we define the overhead for all channels as:

$$O_T = 1 - \frac{U_{DQM}}{U_{DSG}}$$

where U_{DQoM} is the overall link utilization by *DQM* and U_{DSG} is that by *DSG*.

Fig. 9 illustrates the overhead performance for both individual *QoS* channels and overall bandwidth conservation. From the figure we can find that in *DQM* bandwidth for non-gold channels can always be saved and the corresponding overhead varies from 0.33 to 0.46. However, bandwidth for gold channel is not conserved at any time since it cannot be merged into any other *QoS* channel. Regarding the overall bandwidth conservation, from the figure we notice that the aggregated overhead varies from 0.19 to 0.23, i.e., by using *QoS* channel merging in *DQM*, the average bandwidth consumption is 81.3% to 84% that of non-*QoS* merging approaches.

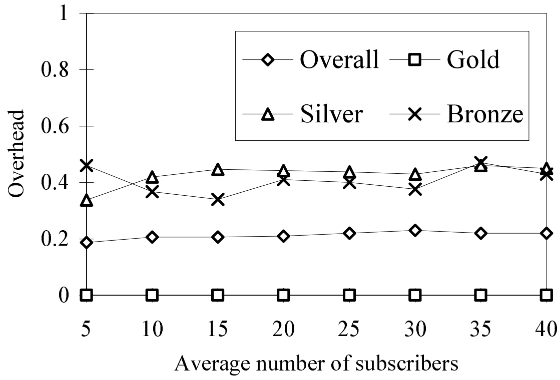


Fig. 9 Overhead of bandwidth conservation

Another interesting empirical study is the traffic engineering capability of *DQM* and *MQ*. In *DQM* network bandwidth is pre-allocated to specific traffic aggregates of individual *QoS* channels, and this is very similar to the strategy of DiffServ. In contrast, *MQ* and *RSVP* allows the overall bandwidth to be accumulatively reserved by various quantities of *QoS* demands until the link has become saturated. In the following simulation, we will examine the performance of load balancing semantics of *DQM* and *MQ/RSVP*. Basing on bandwidth utilization, we classify network links into the following three categories: (1) High load link with overall utilization above 50%; (2) Medium load link with overall utilization between 20% and 50%; and (3) Low load link with overall utilization below 20%. Table 1 presents the proportion of these three types of links inside the network with the average number of subscribers varying from 10 to 50. From the table we can find that *DQM* performs better capability of traffic engineering in that data traffic is more evenly distributed. For example, when the average number of subscribers is below 30, none of the network links become highly loaded by using *DQM*. In contrast, *MQ* always results in hotspots with utilization above 50% even when the average number of subscribers is 10. From the table we can also see that the proportion of low load link in *DQM* is consistently higher than that in *MQ*.

Table 1. Traffic distribution comparison with *MQ*

		10	20	30	40	50
DQM	High load link	0.00%	0.00%	0.00%	0.07%	0.12%
	Medium load link	1.2%	2.6%	4.1%	4.9%	5.4%
	Low load link	98.8%	97.4%	95.9%	95.0%	94.5%
MQ	High load link	0.23%	0.41%	0.86%	1.33%	1.58%
	Medium load link	1.7%	3.1%	4.1%	4.7%	6.7%
	Low load link	98.1%	96.5%	95.0%	94.0%	91.7%

We also investigate the overall link utilization of *DQM* and *MQ*, and the simulation results are presented in Fig. 10. From the figure we notice that the average link utilization of *DQM* is consistently higher than that of *MQ* to a very small scale, e.g., when the average number of subscribers is fixed at 50, the link utilization of *DQM* is 4.7% higher than that of *MQ*. From the empirical results in table 1 and Fig. 10, it can be inferred that *DQM's* better performance of load balancing is in effect at the expense of higher bandwidth consumption, but the relevant cost is very small.

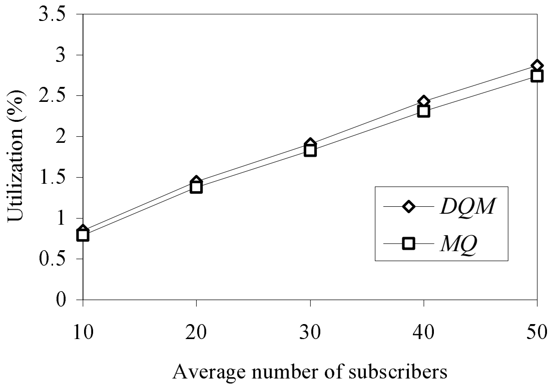


Fig. 10 Overall link utilization comparison with *MQ*

6 Summary

In this paper we proposed a novel scheme called *DQM* that provides differentiated *QoS* channels based on the Source Specific Multicast (*SSM*) service model. This approach efficiently supports heterogeneous *QoS* requirements applications on a qualitative basis. By means of Per channel *QoS* routing and merging mechanism, not only router overhead for maintaining group states is alleviated, but also network bandwidth consumption is reduced compared with traditional solutions such as multicast layered transmission and replicated transmission. Moreover, per *QoS* channel based bandwidth management also contributes to improvements in terms of traffic load distribution compared with the *MQ/RSVP* approaches.

Our future work will address dynamic configuration and management of network resources (e.g., bandwidth pre-emption between *QoS* channels) based on forecasted traffic condition of *QoS* aggregates, and also algorithms for Diffserv-aware multicast traffic engineering as well.

References

1. Blake, S. *et al*: An Architecture for Differentiated Services, *RFC* 2475
2. Bless, R., Wehrle, K.: Group Communication in Differentiated Services Networks, Proc. *IQ2001*, pp. 618-625
3. Braden, R., *et al*: *Resource* ReSerVation Protocol (RSVP) -- Version 1 Functional Specification, RFC 2205, Sept. 1997
4. Cheung, S. *et al*: On the Use of Destination Set Grouping to Improve Fairness in Multicast Video Distribution, Proc. *IEEE INFOCOM'96*, pp553-560
5. Diot, C. *et al*: Deployment Issues for the IP Multicast Service and Architecture, *IEEE* Network, Jan./Feb. 2000, pp 78-88
6. Holbrook, H. W., Cheriton, D. R.: IP Multicast Channels: EXPRESS Support for Large-scale Single-source Applications, Proc. *ACM SIGCOMM'99*
7. Holbrook, H. W., Cain, B.: Source-Specific Multicast for IP, Internet Draft, draft-holbrook-ssm-arch-*.txt, Mar. 2001, work in progress
8. McCanne, S. *et al*: Receiver-Driven Layered Multicast, Proc. of *ACM SIGCOMM'* 96
9. Striegel, A., Manimaran, G.: A Scalable Approach for DiffServ Multicasting, Proc. *ICC'2001*
10. Wang, B., Hou, J.: QoS-Based Multicast Routing for Distributing Layered Video to Heterogeneous Receivers in Rate-based Networks, Proc. *INFOCOM* 2001
11. Waxman, B. M.: Routing of multipoint connections, *IEEE JSAC* 6(9) 1988, pp1617-1622
12. Yang, D. *et al*: MQ: An Integrated Mechanism for Multimedia Multicasting, *IEEE* Trans. on Multimedia, Vol. 3, No. 1., 2001, pp 82-97