

Dynamic Management of QoS for Multimedia Multicasting

Mehul Vora^a and Sunil Kumar^b

^aMath and CS Department, ^bElectrical and Computer Engineering Department

Clarkson University, Potsdam, NY 13699, USA

Email: {voramn, skumar}@clarkson.edu

ABSTRACT

The interaction between multicasting and real-time multimedia streams poses various new and interesting problems in research on communication protocols and architectures. For example, display systems of heterogeneous users in the same session may have different or even varying latency, display resolution, and/or processing capabilities. In this paper, we propose a receiver-oriented resource reservation mechanism, called Dynamic Management of QoS with Priority (DMQP), for multimedia multicasting over the Internet that provides QoS guarantees with service differentiation for heterogeneous users. In DMQP a real-time application requests QoS by specifying a range of bandwidth values and delay, and the network tries to reserve resources for it within its bandwidth range. The service differentiation is achieved by classifying the end-users into normal user and prioritized user. When the number of end-users increases, the new end-users are not simply rejected. Instead, all nodes, including receiver node(s), sender node(s) and intermediate node(s), readjust their reserved resources dynamically to admit more end-users, as long as their minimum bandwidth requirement is met. By treating the bandwidth requirements as ranges, DMQP provides the flexibility needed for operation in the dynamic Internet environment. It has the significant benefit of allowing more flexible sharing of available resources among applications. We have conducted simulations to evaluate the performance of the proposed mechanism.

I. INTRODUCTION

Today, the Internet serves a wide spectrum of applications for huge population of worldwide users, and continues to evolve as a global infrastructure for diverse services. The data traffic carried by the Internet can be classified as: non-real time (such as email and web applications) and real-time (such as streaming and interactive multimedia applications). The *best-effort* service currently supported by Internet handles the non-real time data traffic very well. However, growing number of real-time multimedia applications on Internet, such as video-on-demand, video conferencing, voice-over-IP, visualization and virtual reality, require more stringent quality of service (QoS) guarantees, in terms of available bandwidth, reliable and delay-bound delivery of data, to one or more users simultaneously [1]-[5]. Communication of data from one-to-many and many-to-many recipients by resource sharing (to avoid sending separate packets from sender to multiple receivers) in Internet environment is known as IP-multicasting. In order to provide this capability multicasting and QoS guarantees must be incorporated into the network infrastructure [6]-[8].

QoS support for real-time multimedia multicast applications requires some kind of resource reservation, in conjunction with QoS routing. Otherwise, data may be lost or delayed due to resource shortage, when network is heavily loaded. RSVP (Resource reSerVation Protocol) is a widely used receiver-oriented reservation protocol to reserve network resources [9]-[11]. However, the

integration of RSVP with QoS multicast routing protocols, such as multicast routing extensions for open shortest path first (MOSPF) and QoS extension to OSPF (QOSPF), makes it ‘*sender-oriented*’ [12]. Recently, a truly ‘*receiver-oriented*’ mechanism, known as MQ (Multicast with QoS), was proposed for multimedia multicasting. MQ integrates IP multicasting, QoS routing, and resource reservations, in the presence of user-heterogeneity, while making efficient use of network resources [12]. Please note that as compared to sender-oriented reservation schemes, receiver-oriented reservation can more effectively accommodate user heterogeneity and shows superior scaling property. Here user heterogeneity means that receivers in the same session have different or even varying latency, display resolution, and/or processing capabilities.

In RSVP as well as MQ, a receiver specifies its QoS requirements (primarily fixed-bandwidth and delay) to the network. However, problem with the fixed bandwidth requirement is that the network can provide only the “best or nothing” service. In many multimedia applications, it is acceptable to get the connection at a lower service level (e.g., reduced bit-rate), rather than not getting connection at all. The newly developed compression standards, such as MPEG-4 and JPEG2000, use layered and fine-granular scalability where the bitstream can be decoded with whatever data of a flow is available. Even readjustment of reserved resources for the existing receivers, to accommodate more new receivers, may be acceptable in many multimedia applications. Very recently, an extension of RSVP, known as Dynamic RSVP (DRSVP), has been proposed, which allows a receiver to specify its bandwidth range, instead of a fixed bandwidth [13]. All three schemes, RSVP, DRSVP and MQ, differentiate the real-time data traffic from non-real time data traffic and provide better QoS support for real-time data by the means of resource reservations. But within the class of real-time data, there is no provision for service differentiation in terms of priority. User priority may play a very important role in applications such as telemedicine or emergency management. ISPs may also use the service differentiation as a way to obtain higher revenue.

To support dynamic QoS with user priority, we propose a receiver-oriented resource reservation mechanism that we call as *Dynamic Management of QoS with Priority (DMQP)*. DMQP is inspired by the concepts used in MQ [12] and DRSVP [13]. In DMQP, a receiver requests QoS by specifying a bandwidth range, delay and priority level, and the network tries to support its QoS requirements by reserving appropriate resources. Besides DMQP provides a more flexible mechanism to dynamically adjust the reserved resources on nodes, including receiver node(s), sender node(s) and intermediate node(s) along the reserved path.

The rest of the paper is organized as follows. Section 2 provides a detailed description of the proposed DMQP mechanism, in terms of setup and maintenance processes. In Section 3, we evaluate and compare the performance of DMQP. Finally concluding remarks are presented in Section 4.

II. DYNAMIC MANAGEMENT OF QOS WITH PRIORITY (DMQP)

DMQP is a *unidirectional, session-specific*, and truly *receiver-oriented* dynamic reservation approach to provide end-to-end QoS guarantee as discussed below.

1. Application requests QoS by specifying a range of values (Q_{min} , Q_{max}) where Q_{min} is the minimum level of service the application is willing to accept and Q_{max} is the maximum level of service that the

application is able to utilize. Bandwidth is a fundamental network resource, as its allocation determines the applications' maximum throughput and, in some cases, the bounds on end-to-end delay. Therefore, in our scheme, while making and handling a reservation request, we mainly focus on bandwidth requirements.

2. Service differentiation is achieved by associating a priority with some end users. Assignment of priority is based on Service Level Agreement (SLA) between end users and their Internet Service Providers (ISPs), as is the case in the DiffServ architecture. Here we differentiate the receivers of real-time flow into 'normal' and 'prioritized' classes.

DMQP mechanism can be divided in two parts: setup process and maintenance process. Some messages such as *FlowAd*, *JoinRequest*, used in DMQP are adopted from MQ [12]. However, their handling is quite different, in order to support dynamic QoS.

2.1. Setup Process:

A source advertises its *flowspec* (maximum peak rate and token bucket depth) by sending a *FlowAd* message to all group members, using shortest path multicast routing algorithm. The receiver responds to it by sending a *JoinRequest* message specifying its QoS requirements in terms of its bandwidth range (R_{min} , R_{max}) and delay, based on sender's *flowspec* and any local knowledge. Here we would like to introduce the notion of $R_{acceptable}$, $R_{requirement}$ and $R_{borrowable}$ bandwidths. $R_{acceptable}$ is the bandwidth required by applications to perform at a reasonably good quality. We define it as the midpoint of (R_{min} , R_{max}). $R_{requirement}$ is the actual bandwidth demand to the network for reservation. It is a network dependent parameter. Receiver(s) send the *JoinRequest* message with R_{max} as the $R_{requirement}$. While forwarding the message upstream through the network, an intermediate router may change the $R_{requirement}$, depending upon the available link bandwidth. This is explained in Section 4. When the bandwidth reserved for a flow is more than $R_{acceptable}$, their difference is defined as $R_{borrowable}$. The network can borrow this bandwidth from an existing flow in order to accommodate a new flow in case of scarcity of resource.

Upon receiving a *JoinRequest* message, the intermediate routers on the path towards source record the path states and temporarily reserve the resources for it. While forwarding *JoinRequest* messages upstream, an intermediate router merges all the requests for the same flow, and forwards a common *JoinRequest* message with the (possibly modified) *flowspec*. A *JoinRequest* message travels upstream until the requested reservation level is met. From there, a *JoinAck* message is returned following the same path in the reverse direction, confirming the reservation in the intermediate routers.

If sufficient bandwidth is available to accommodate a new receiver at its $R_{acceptable}$ or higher bandwidth, it is simply admitted. Otherwise, the network tries to reserve bandwidth in the range of (R_{min} , $R_{acceptable}$), by borrowing the required bandwidth from the existing flows in the following order: First the $R_{borrowable}$ bandwidth is borrowed from the normal users. If this is not sufficient and the new user is a prioritized user, $R_{borrowable}$ is borrowed from other prioritized users. If this is also not sufficient, then bandwidth is borrowed from normal users above their R_{min} .

In case of a very heavily loaded network, where it not possible for the network to support even R_{min} requirement, a *JoinFail* is sent to the immediate downstream router, from where the *JoinRequest* came. Upon receiving a *JoinFail* message, this router acts as a breakout router. A router also acts as a breakout router when it receives a *JoinAck* message with allocated bandwidth that is less than the $R_{requirement}$. The break out router determines a new feasible path with sufficient resources, using QoS routing. If such a

path is found, the router forwards the *JoinRequest* toward the new path and waits for an acknowledgment. Upon receiving a *JoinAck*, the breakout router forwards the *JoinAck* downstream to confirm the reservation, and sends a *ResvRemove* (reservation remove) message upstream in the old path to relinquish previously reserved resources.

The *ResvRemove* is also sent by the receiver while ending its session. If a router detects that the departing interface (i.e., the interface where the *ResvRemove* message came from) has the highest reserved bandwidth, it sends a *Shrink* message to upstream link. The *Shrink* message reduces the reserved bandwidth for the corresponding flow, such that the maximum reserved bandwidth required by all the existing downstream interfaces of this flow is still satisfied.

2.2. Maintenance Process:

Maintenance refers to the operations that are conducted on an established DMQP session based on the network dynamics - mainly link failures and topology changes, joining and/or leaving of users, and changes in the requested QoS of existing users.

To reserve the resources for joining an ongoing session, a new user must have *flowspec* of the source(s). A receiver can acquire sources' *flowspec* by waiting for their *FlowAd* message that is periodically sent to all the receivers who have joined the group. Alternatively, a receiver can request *FlowAd* by sending a *FlowSolicit* message to the relevant source(s). On receiving *FlowSolicit*, sources send *FlowAd* to all the participants of the group.

When a network router receives a *ResvRemove/Shrink* message from a receiver leaving the group, it frees the bandwidth reserved for it. The network router also notifies other receivers of the flow(s) that are currently getting service at less than their acceptable and/or maximum demand, about the availability of bandwidth. The interested receiver(s) will respond by resending *JoinRequest* message.

A change in receiver(s) bandwidth requirement is handled by resending *JoinRequest* message. If there is any change in source characteristics, source sends a new *FlowAd* message to all the group participants notifying them about the changes. Based on this new *FlowAd*, the receiver(s) send a new *JoinRequest* message, if necessary.

The network dynamics, such as link failure and topology changes, are handled by *Refresh* message. After the route has been established and resources successfully reserved along the established route, the receivers must periodically send a *Refresh* message upstream. The main functions of *Refresh* messages are: a) to refresh the *soft-state* (timer) associated with the flow in each intermediate router, and b) to respond to any topology change. If a router does not receive a *Refresh* message before the expiration of timer, it assumes the downstream receiver(s) no longer exist and issues a *ResvRemove* message to release the previously reserved resources.

III. PERFORMANCE EVALUATION

In this section we will evaluate the performance of proposed DMQP mechanism using Network Simulator - Version 2.26 (ns2) package [14].

3.1. Simulation Setup:

The network topology was generated using Boston University Representative Internet Topology generator (BRITE) [15], which provides a wide variety of generation models. Network topology uses Flat graph model along with Waxman's probability model for interconnecting the nodes. In this model the nodes are distributed randomly in the plane and the probability for an edge between any pairs of nodes (u, v) is given by:

$$P(u, v) = \alpha e^{-\beta L} \quad \dots(1)$$

where $\alpha, \beta \in (0, 1)$, d is the Euclidean distance between nodes u and v , and L is the maximum distance between any two nodes. An increase in α (β) increases the edge densities (number of connections of distant nodes). Results presented here are generated using a 100-node network with α and β values set to 0.15 and 0.2, respectively. Bandwidth assigned to the link between node u and node v is given by:

$$bw(u, v) = \text{rand}(B1, B2) \quad \dots(2)$$

where $B1 = 128$ Kbps, $B2 = 3088$ Kbps, and $\text{rand}(B1, B2)$ denotes a random number between $B1$ and $B2$. Each simulation run was conducted for 120 minutes. The inter-arrival time was exponentially distributed with mean of 4 minutes, and the time a receiver spends in a session was exponentially distributed with a mean of 40 minutes. While determining the path for *JoinRequest* message, QoS routing algorithm considers hop count as delay requirements. Table 1 represents the traffic composition used in the simulation. In DMQP, we set R_{min} to 50% of R_{max} for a receiver, except for the traffics belonging to 32Kbps flow, which we treat as constant bit rate stream. Since MQ scheme does not use service differentiation, the first and last columns of Table 1 represent its traffic composition, with R_{max} as fixed bit rate.

Table 1: Traffic Composition

| R_{max} (Kbps) | Weight (K) | % of total receivers | | |
|---------------------|---------------|----------------------|----------------|------------|
| | | Normal Users | Priority Users | Total |
| 32 | 1 | 17.25 | 5.75 | 23 |
| 64 | 2 | 26.25 | 8.75 | 35 |
| 128 | 4 | 18.75 | 6.25 | 25 |
| 256 | 8 | 9.00 | 3.00 | 12 |
| 512 | 16 | 3.75 | 1.25 | 5 |
| Total | | 75.00 | 25.00 | 100 |

3.2. Simulation Results:

It was shown in [12] that MQ has lower blocking probability, higher resource utilization, and lower normalized resource consumption as compared to QoS routing and resource reservation-based schemes, such as RSVP+MOSPF and RSVP+QOSPF. Therefore, we compare the performance of DMQP with MQ alone.

Weighted Blocking Probability:

When a user is allocated resources below its R_{max} , at any time during the session, we consider it as partially blocked. Taking partial blocking into consideration, the weighted blocking probability is given by:

$$\text{Weighted } P_B = \frac{\sum K_i * [0.25(A_i) + 0.5(M_i) + B_i]}{\sum K_i * (P_i + A_i + M_i + B_i)} \quad \dots(3)$$

where, P_i , A_i and M_i represent the number of users in flow i having service at R_{max} , ($R_{acceptable}$, R_{max}) and (R_{min} , $R_{acceptable}$), respectively. B_i is the number of blocked users and K_i is the weight associated with the flow i (proportional to the R_{max} of the flow) as given in Table 1. We consider the bandwidth allocated to a user as $R_{acceptable}$ (R_{min}), even if it was mostly served at a bandwidth of R_{max} ($R_{acceptable}$), but its bandwidth dropped below R_{max} ($R_{acceptable}$) (but above or equal to $R_{acceptable}$) only for a short time during the session. Note that A_i and M_i are set to zero in MQ, as there are no partially blocked users.

As shown in Fig. 1, the weighted blocking probability for DMQP and MQ protocols increases with the number of sessions. DMQP has lower weighted blocking probability than MQ, because it minimizes the unused link bandwidth by readjusting the bandwidth allocations to admit more users at lower bandwidth.

Resource Utilization:

Resource utilization is defined as the ratio of actually used bandwidth to the total link bandwidth. Since the resource consumption increases with the number of admitted receivers, we normalize it with the total number of admitted receivers, as in MQ [12].

As shown in Fig. 2, the resource utilization of DMQP and MQ schemes increases with number of sessions. However, DMQP more efficiently uses available resources because network can readjust the reserved bandwidth to admit more receivers and thus minimize the unused bandwidth. For the same reason, DMQP consumes less resource per admitted user (i.e., normalized resource consumption) as compared to MQ, in moderate to heavily loaded network, as shown in Fig. 3.

Performance Benefit:

Performance benefit represents the overall degree of end-users' satisfaction and network performance. We define it as:

$$\text{Benefit} = \sum_{i=1}^{n1} K_j * \frac{\text{Minimum bandwidth reserved during the session}}{\text{Maximum bandwidth requirement}} + \lambda \sum_{i=1}^{n2} K_j * \frac{\text{Minimum bandwidth reserved during the session}}{\text{Maximum bandwidth requirement}} \quad (4)$$

where $n1$ and $n2$ are the total number of normal and prioritized users, respectively; λ is the weighting factor for prioritized users, and K_j is the weight associated with flow j as given in Table 1.

Lower blocking probability and higher resource utilization implies greater performance benefit for DMQP scheme. Since there is no priority class in MQ, we set λ to 1 for a fair comparison with MQ. Fig. 4 shows that DMQP, indeed, has better performance benefit than MQ scheme. Setting λ to 1 is, however, very conservative as it does not reflect the benefit of accommodating prioritized receivers in DMQP.

IV. CONCLUSION

We proposed an improved receiver-oriented bandwidth reservation mechanism, DMQP (Dynamic Management of QoS with Priority) that provides QoS guarantees with service differentiation for heterogeneous users, in multimedia multicasting applications over the Internet. The DMQP differs from other resource reservation protocols, such as RSVP [10-11] and MQ [12], in the following three aspects. First, a real-time application requests bandwidth by specifying a range of values, instead of a fixed value. Second, DMQP differentiates the end-users into two service classes: normal user and prioritized user, rather than treating all the end-users equally. Third, when the number of end-users competing for resources increases, the new end-users are not simply rejected. Instead, all nodes, including receiver node(s), sender node(s) and intermediate node(s), readjust their reserved resources dynamically to admit more end-users, as long as their minimum bandwidth requirement is met, depending upon the service class.

By treating the bandwidth requirements as ranges, DMQP provided the flexibility needed for operation in a dynamic environment. It has the significant benefit of allowing more flexible sharing of available resources among applications. Our experience in simulating this adaptability has convinced us that a dynamic QoS approach is both feasible and potentially valuable. By adopting dynamic reservation approach, DMQP demonstrated lower blocking probability, more efficient resource utilization and less resource consumption per admitted user, and better performance benefit as compared to MQ [12]. Besides, it maintains other properties of MQ, such as scalability, robustness and loop-free control.

REFERENCES

- [1]. G. Lu, "Issues and technologies for supporting multimedia communications over the Internet", *Computer Communications*, Vol. 23 (14-15), pp. 1323-1335, Aug. 2000.
- [2]. D. Wu, Y. Hou, Y. Zhang, "Transporting real-time video over the Internet: challenges and approaches", *Proceedings of IEEE*, Vol. 88(12), pp. 1855-1877, Dec. 2000.
- [3]. W. Zhao, D. Olshefski, H. Schulzrinne, "Internet quality of service: an overview", Technical Report, CUCS-003-00, Columbia University, Computer Science Department, Feb. 2000.
- [4]. A. Hafid, G. Bochmann, R. Dssouli, "distributed multimedia application and quality of service: a review", *Electronic Journal on Networks and Distributed Processing*, pp. 1-50, Feb. 1998.
- [5]. V. Firoiu, J. Boudec, D. Towsley, "Theories and models for Internet quality of service", *Proceedings of IEEE*, Vol. 90(9), pp. 1565-1591, Sept. 2002.
- [6]. A. Striegel, G. Manimaran, "A survey of QoS multicasting issues", *IEEE Communication Magazine*, pp. 82-87, June 2002.
- [7]. J. Pasquale, G. Polyzos, G. Xylomenos, "The multimedia multicasting problem", *Multimedia Systems*, Vol. 6(1), pp. 43-59, 1998.
- [8]. A. Vogel, B. Kerherve, G. Bochmann, J. Gecsei, "Distributed multimedia applications and quality of service: a survey", *IEEE Multimedia Journal*, Aug. 1995.
- [9]. L. Zhang, S. Deering, D. Estrin, S. Shenker, D. Zappala, "RSVP: a new Resource reSerVation Protocol", *IEEE Network Magazine*, pp. 8-18, Sept. 1993.
- [10]. R. Braden, L. Zhang, S. Berson, S. Herzog, and S. Jamin, "Resource reSerVation protocol (RSVP)—version 1 functional spec", *IETF RFC 2205*, Sept. 1997.
- [11]. J. Wroclawski, "The use of RSVP with IETF integrated services", *IETF RFC 2210*, Sept. 1997.

- [12]. D. Yang, W. Liao, "MQ: an integrated mechanism for multimedia multicasting", IEEE Trans. Multimedia, Vol. 3(1), pp. 82-97, March 2001.
- [13]. G. Kuo, P. Ko, "Dynamic RSVP protocol", IEEE Communication Magazine, Vol. 41(5), pp. 130-135, May 2003.
- [14]. K. Fall, K. Varadhan, "The ns manual", (<http://www.isi.edu/nsnam/ns>)
- [15]. A. Medina, A. Lakhina, I. Matta, J. Byers, "BRITE: Universal Topology Generation from a User's Perspective", Technical Report BUCS-TR2001-003, Boston University, 2001. Available at <http://www.cs.bu.edu/brite/publications>.

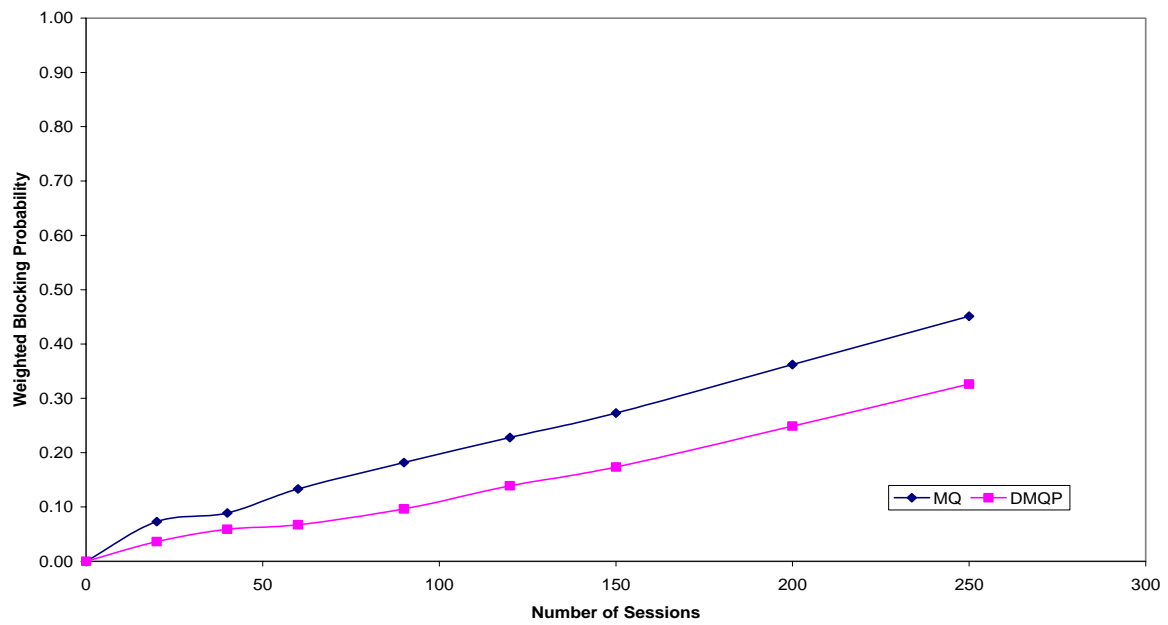


Fig. 1: Weighted blocking probability

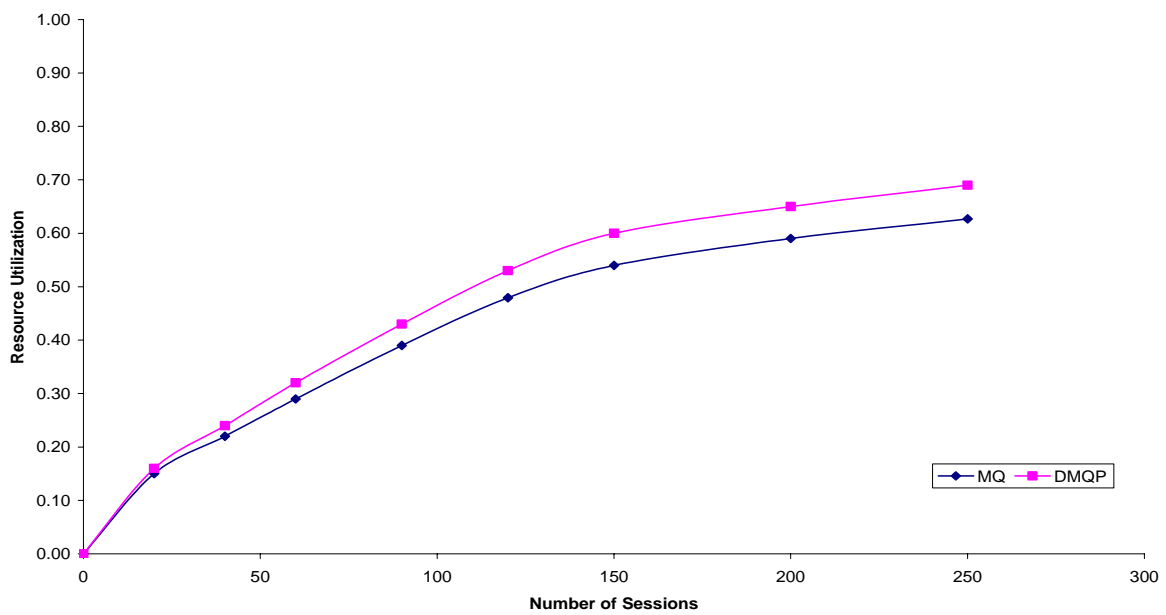


Fig. 2: Resource Utilization

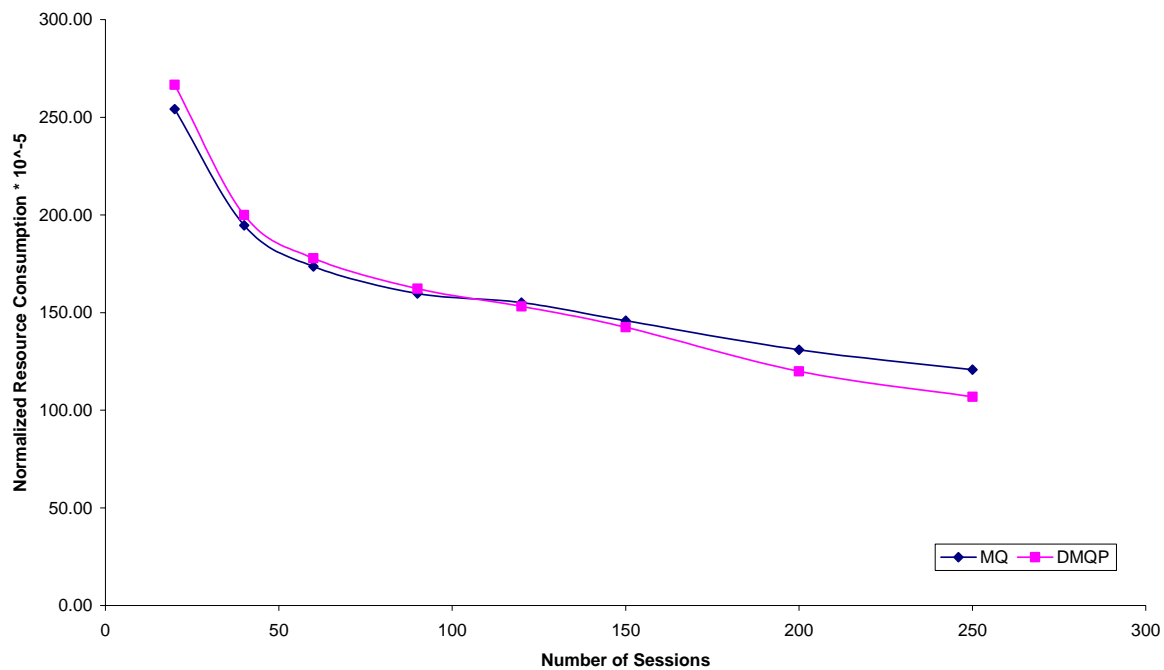


Fig. 3: Normalized resource consumption

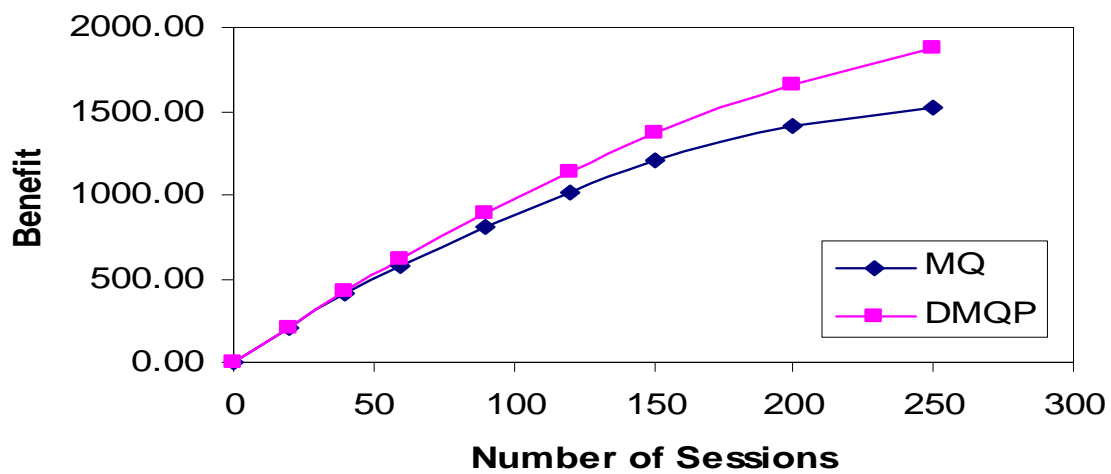


Fig. 4: The performance benefit