LUND UNIVERSITY

Ad Hoc quality of service multicast routing

Bür, Kaan; Ersoy, Cem

Published in:
Computer Communications

DOI:
10.1016/j.comcom.2005.01.016

2005

Document Version:
Peer reviewed version (aka post-print)

Link to publication

Citation for published version (APA):
Bür, K., & Ersoy, C. (2005). Ad Hoc quality of service multicast routing. *Computer Communications*, *29*(1), 136-148. https://doi.org/10.1016/j.comcom.2005.01.016

Total number of authors:
2

General rights
Unless other specific re-use rights are stated the following general rights apply:
Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.
• Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
• You may not further distribute the material or use it for any profit-making activity or commercial gain
• You may freely distribute the URL identifying the publication in the public portal

Read more about Creative commons licenses: https://creativecommons.org/licenses/

Take down policy
If you believe that this document breaches copyright please contact us providing details, and we will remove access to the work immediately and investigate your claim.

LUND UNIVERSITY

PO Box 117
221 00 Lund
+46 46-222 00 00

Ad Hoc Quality of Service Multicast Routing

Kaan Bür[1], Cem Ersoy

NETLAB, Department of Computer Engineering
Bogaziçi University, 34342 Bebek, Istanbul, Turkey
{burk, ersoy}@boun.edu.tr

Abstract

The conceptual shift in expectations of wireless users from voice towards multimedia, from availability towards acceptable quality, and from stand-alone towards group-oriented computing has a significant impact on today's networks in terms of the need for mobility, quality of service (QoS) and multicasting. Ad hoc networks, being independent of any fixed infrastructure, can provide mobile users with these features, if necessary QoS multicasting strategies are developed. The aim of this article is to define the building blocks of such an ad hoc QoS multicasting (AQM) protocol. AQM achieves multicasting efficiency by tracking the availability of resources for each node within its neighbourhood. Computation of free bandwidth is based on reservations made for ongoing sessions and similar information reported by neighbours. Current QoS status is announced at the initiation of a new session and updated periodically in the network to the extent of QoS provision. Thus, nodes are prevented from applying for membership if there is no QoS path for the session. When nodes join a session with certain service requirements, a request-reply-reserve process ensures that the QoS information is refreshed and used to select the most appropriate routes. To evaluate the efficiency of AQM in providing multicast users with QoS and satisfying application requirements, two new performance metrics, member and session satisfaction grades are introduced. AQM is compared to a non-QoS scheme with particular emphasis on these criteria. Simulation results show that, by applying QoS restrictions, AQM significantly improves multicasting efficiency. Thus, QoS is both essential for and applicable to multicasting in order to support mobile multimedia applications in ad hoc networks.

Keywords: Ad hoc networks, mobile multimedia, multicast routing, quality of service, wireless communication.

1. Introduction

THE increasing popularity of video, voice and data communications over the Internet, and the rapid penetration of mobile telephony have stimulated a change in the expectations of wireless users. The number of group-oriented services and multimedia applications is increasing. Research and development are taking place to define the next generation of wireless broadband multimedia communication systems. The internetwork of the future will probably consist of a fixed network with a wired backbone, an infrastructured mobile network with base stations, and at the peripherals, ad hoc mobile networks, which will be connected to the main internetwork via ad hoc switches [1]. While current communication systems are primarily designed for one specific type of application such as speech, video or data, the next generation will integrate various functions and applications. Therefore, it is essential that wireless and multimedia be brought together [2], which necessitates acceptable quality instead of mere availability.

Ad hoc networks are impromptu communication groups formed by wireless mobile hosts without any established infrastructure or centralised control, which are becoming increasingly popular as a result of these developments. They are considered for many commercial applications, including in-home networking, wireless local area networks, nomadic computing, and short-term communication for disaster relief, public events, and temporary offices. In this regard, ad hoc networks have to support multimedia applications, which make quality of service (QoS) a critical issue. QoS defines a guarantee given by the network to satisfy a set of predetermined service performance constraints for the user in terms of end-to-end delay, jitter, available bandwidth, and packet loss probability [3]. QoS support for multimedia applications is closely related to resource allocation, the objective of which is to decide how to reserve resources such that QoS requirements of all the applications can be satisfied [4]. However, it is a significant technical challenge to provide reliable high-speed end-to-end communications in ad hoc networks, due to their dynamic topology, distributed management, and multihop connections [5]. Thus, it becomes very important to utilise scarce resources effectively in this unreliable environment. Multicasting is a promising technique to provide a subset of network nodes with the service they demand while not jeopardizing the bandwidth requirements of others. The advantage of

Manuscript submitted to Computer Communications on July 15, 2004; revised version submitted on January 18, 2005.

A preliminary version of this work is presented at PIMRC 2004 – the 15th IEEE International Symposium on Personal, Indoor and Mobile Radio Communications, Barcelona, Spain, September 5-8, 2004.

This work was supported in part by the State Planning Organisation, Turkey, under grant numbers DPT98K120890 – DPT03K120250 and the university research program of OPNET Technologies Inc, USA.

[1] Corresponding author. *Tel:* +90-212-359-7170. *Fax:* +90-212-287-2461.

multicasting is that packets are only multiplexed when it is necessary to reach two or more receivers on disjoint paths. As a result of their broadcasting capability, the nodes of an ad hoc network are inherently ready for multicasting.

It is not an easy task to incorporate QoS to ad hoc multicasting. Incremental changes on existing schemes cannot efficiently address the critical issues mentioned above. In this paper, the ad hoc QoS multicasting (AQM) protocol is presented as a composite solution to the problem. AQM tracks QoS availability within each node's neighbourhood based on current reservations, and announces it at session initiation. When a node wants to join a session with certain service requirements, a request-reply-reserve process ensures that this QoS information is updated and used to select one of the routes which can meet the requirements of that session. Simulation results show that AQM significantly improves multicasting efficiency for members and sessions through QoS management. The rest of this paper is organised as follows. Previous research related to ad hoc multicasting is summarised in Section 2. AQM is introduced in Section 3. The performance of the proposed system is evaluated in Section 4. Concluding remarks and future work are presented in Section 5.

2. An Overview of Ad Hoc Multicasting

There are various protocols developed to build and maintain a multicast graph and perform routing in ad hoc networks, some of which are summarised below. However, they do not attempt to cover the QoS aspect of ad hoc multicast communication, which is becoming increasingly important as the demand for mobile multimedia increases.

Multicast ad hoc on demand distance vector (MAODV) routing protocol is derived from AODV [6, 7]. The multicast group leader maintains a group sequence number and broadcasts it periodically to keep the routing information fresh. A node wishing to join a multicast group generates a route request. If the multicast group leader is in the request table, the request is unicast to it. Otherwise, the request is broadcast. Only the leader or members of the multicast group may respond to a join request by unicasting a route reply back to the requester. Other nodes may only rebroadcast or forward the packet. Nodes receiving join requests update their route and multicast tables with the downstream next hop information. Nodes receiving reply messages update their tables with the upstream next hop information. They increment hop counts and forward the reply to the requester, which selects the best from several replies in terms of highest sequence numbers and lowest hop count, and enables that route by unicasting a multicast activation message to its next hop. Intermediate nodes receiving the activation message enable their multicast table entries for the requester. If they are already multicast group members, further propagation of the message is not necessary. Otherwise, they unicast it upstream along the best route according to the replies they received previously. Nodes wishing to leave a group unicast a multicast activation message to their next hop with its prune flag set.

The core-assisted mesh protocol (CAMP) uses cores within a group to limit the control traffic caused by join requests [8, 9]. Each node defines its predecessor in the multicast mesh from which it receives data as its anchor. In other words, anchors are those nodes that are expected to rebroadcast the multicast data they receive to the routers downstream. Each node maintains a set of tables for routing, core-to-group mapping, and anchor and group management. When a node updates its anchor or multicast table, it sends a reporting message to all its neighbours. The basic join mechanism is initiated by a host asking its router to join a group. The router directly announces its membership if there are any data-forwarding members of that group among its neighbours. Otherwise, it broadcasts a join request. Member routers of the intended group send acknowledgements. The requesting router and its relays become part of the group as soon as they receive the first acknowledgement. A router leaves a multicast group if it has no member hosts and is not required as an anchor.

Bandwidth-efficient multicast routing (BEMR) finds the nearest forwarding multicast member for newly joining nodes [10]. When a new node broadcasts a join request, each node receiving the request adds its ID and increments the hop count before flooding it back to the network. Forwarding nodes receive some of these requests, choose the best hop alternative and send a reply packet along the selected path. The requester eventually receives multiple replies, chooses the best hop alternative and sends a reserve packet along the same path. All nodes on this path become forwarding nodes. Routes are later optimised by removing unnecessary forwarding nodes.

The on-demand multicast routing protocol (ODMRP) introduces the concept of a forwarding group [11, 12]. Sources periodically broadcast join query messages to invite new members and refresh existing membership information. When a node receives a join query, it stores the upstream node address in its routing table. If the maximum hop count is not exceeded, it updates the join request using this table and rebroadcasts the packet. When a node decides to join a session, it broadcasts a join reply. When a node receives a join reply, it checks the table of next nodes to see if it is on the path to the source. If this is the case, it sets its forwarding group flag and broadcasts its own join reply after updating the table of next nodes. Periodic join requests initiated by the source must be answered by session members with join replies to remain in the group. Forwarding group nodes reset their flags if they do not receive any replies periodically.

Neighbour-supporting multicast protocol (NSMP) utilises node locality to reduce route maintenance overhead [13]. A mesh is created by a new source, which broadcasts a flooding request. Intermediate nodes cache the upstream node information contained in the request, and forward the packet after updating this field. When the request arrives at

receivers, they send replies to their upstream nodes. On the return path, intermediate nodes make an entry to their routing tables and forward the reply upstream towards the source. In order to maintain the connectivity of the mesh, the source employs local route discoveries by periodically sending local requests, which are only relayed to mesh nodes and their immediate neighbours to limit flooding while keeping the most useful nodes informed. Replies are sent back to the source to repair broken links. Nodes more than two hops away from the source cannot join the mesh with local requests. They have to flood member requests.

Associativity-based ad hoc multicast (ABAM) builds a source-based multicast tree [14]. Association stability, which is achieved when the number of beacons received consecutively from a neighbour reaches a threshold, helps the source select routes which will probably last longer and need fewer reconfigurations. The tree formation is initiated by the source, whereby it specifically identifies its receivers. Valid receivers, which already know possible routes to the source, run a route selection algorithm to select and reply with routes of highest association stability. Upon receiving the replies, the source runs a tree selection algorithm to find common links, builds the shared-link multicast tree, and sends a setup message to its receivers. Tree reconfigurations occur when the associative property is violated. To join a multicast tree, a node broadcasts a request, collects replies from group members, selects the best route, and sends a confirmation. To leave a multicast tree, a notification is propagated upstream along the tree until a branching or receiving node is reached.

Differential destination multicast (DDM) lets source nodes manage group membership as admission controllers, and stores multicast forwarding state information encoded in headers of data packets to achieve stateless multicasting [15, 16]. Join messages are unicast to the source, which tests admission requirements, adds the requester to its member list, and acknowledges it as a receiver. The source needs to refresh its member list in order to purge stale members. It sets a poll flag in data packets and forces its active receivers to resend join messages. Leave messages are also unicast to the source, which removes the leaving member from its list. Forwarding computation is based on destinations encoded in the headers. During this process, a node has to check the header for any DDM block or poll flag intended for it and take the appropriate actions.

Independent-tree ad hoc multicast routing (ITAMAR) provides several heuristics to compute a set of independent multicast trees, such that a tree is used until it fails and then replaced by one of its alternatives [17]. Maximally independent trees are computed by minimising the number of common edges and nodes under the assumption that node movements are independent of each other. Some overlapping is allowed since totally independent trees might be less efficient and contain more links. Thus, the correlation between the failure times of the trees is minimal, which leads to improved mean times between route discoveries. New trees are computed when the probability of failure for the current set of trees rises above a threshold. Given a mobility pattern, it is important to estimate the time this happens. Instead of replacing a tree even if one link fails, an independent path algorithm finds a set of backup paths to replace the damaged part of the tree.

Lantern-tree-based QoS multicast (LTM) is a bandwidth routing protocol with an improved success rate by means of multipath routing [18, 19]. A lantern is defined as one or more subpaths with a total bandwidth between a pair of two-hop neighbouring nodes. A lantern path is a path with one or more lanterns between a source and a destination. Finally, a lantern tree is defined as a multicast tree which contains at least one lantern-path between any of its source-destination pairs. The scheme provides a single path if bandwidth is sufficient or a lantern-path if it is not. The replying paths from the destination back to the source are merged together to construct the lantern tree.

Probabilistic predictive multicast algorithm (PPMA) tracks relative node movements and statistically estimates future relative positions to maximise the multicast tree lifetime by exploiting more stable links [20]. Thus, it tries to keep track of the network state evolution. It defines a probabilistic link cost as a function of energy, distance and node lifetime. The scheme tries to keep all the nodes alive as long as possible. It models the residual energy available for communication for each node, which is proportional to the probability of being chosen to a multicast tree. Nodes of low energy cannot join any more multicast trees. The algorithm has a centralised and a distributed version.

Research on combining QoS with ad hoc networks does not address multicasting and is mainly limited to routing. Existing work generally assumes a time division multiple access (TDMA) or a clustered code division multiple access (CDMA) over TDMA network synchronised on a frame and slot basis, where topologies do not change very fast and slot assignment is left to the underlying medium access control (MAC) layer. If the channel access scheme is CDMA on top of TDMA as in [21], there is a control part in each frame, where nodes exchange connectivity information and clusterheads assign slots and code to virtual circuit (VC) requests. In [22], a set of free and non-conflicting slots is calculated on three adjacent links and propagated towards the destination. Bandwidth calculation is done end-to-end, i.e., only destinations can reply to requests. In [23], imprecise QoS information is kept at each node for every other, whereas information on immediate neighbours is kept more accurately. MAC assumptions include a mechanism of beacons, contention resolution, and local message broadcasting. In [24], sources are informed on QoS to any destination. A VC mechanism is implemented at call setup to enable a fast reservation scheme where TDMA slots are reserved to VCs. In [25], each node broadcasts its QoS information during the control phase, at the end of which each node knows the channel reservation status of the next information phase.

3. The Ad Hoc QoS Multicasting Protocol

The motivation behind QoS support for multicasting in ad hoc networks is the fact that mobile multimedia applications are becoming increasingly important for group communication. For an efficient ad hoc QoS multicasting strategy, implementation of QoS classes, negotiations between the network and its users, bounded loss and delay, bandwidth reservation, and mobility management are very important. In the following sections, the structural components of AQM are defined, which address these issues. Design details include usage of QoS classes, initiation and termination of multicast sessions, membership management, allocation of resources, and neighbourhood maintenance.

3.1. Usage of QoS Classes

Different QoS classes are necessary to support various types of applications in an efficient manner. In any multimedia network, there may be multiple application types being run simultaneously, which need to be classified in terms of their varying QoS requirements. To represent such a generic networking environment in this work, a sample set of multimedia applications are suggested. Depending on the user profiles, network conditions and computational capabilities of the mobile multimedia devices, other applications with different QoS settings can easily be added to the set.

Defining QoS classes also limits the amount of information to be transmitted between network nodes. It is otherwise impossible to define and forward a best QoS combination without making some assumptions or disregarding some valuable data about the current QoS conditions being experienced by the network. Therefore, it may be preferable that nodes only inform others on the availability of a certain QoS support level and send updates only when this level changes.

3.2. Session Initiation and Termination

A session is started by a session initiator (MCN_INIT), which can be any node that broadcasts a session initiation packet (SES_INIT) consisting of the identity number and the QoS class of the new session, which sets the bandwidth and hop count rules to join it. If necessary, the session definition can be extended with the duration and the cost of the application, the minimum number of users to activate the session, and the maximum number of acceptable users. A table of active sessions (TBL_SESSION) is maintained at each node to keep the information on session definitions. Figure 1 shows the phases of session initiation.

Using their session tables, nodes forward initiation packets of previously unknown sessions. A membership table (TBL_MEMBER) is used to denote the status of the predecessors (MCN_PRED) which have informed the node on the existence of a particular multicast session, and the QoS support status of the path from the session initiator up to this node via that predecessor. The hop count information in the packet is used to prevent loops in the forwarding process. The session initiation packet is forwarded as long as the QoS requirements are met. Before the packet is rebroadcast, each node updates its QoS information fields with the current QoS conditions experienced by that node. The packet is dropped if QoS requirements cannot be met any more to avoid flooding.

The session information is refreshed periodically via session update packets (SES_UPDATE) sent by the session initiator. Similar to the session initiation packets, they are propagated throughout the network as long as the QoS requirements of the session can be fulfilled. Unlike the session initiation packets, however, each new update packet is forwarded once even if it belongs to a previously known session and come from a known predecessor. This is important since the aim of the session update packets is to ensure that all new nodes in a neighbourhood are informed on the existence of the ongoing sessions they can join.

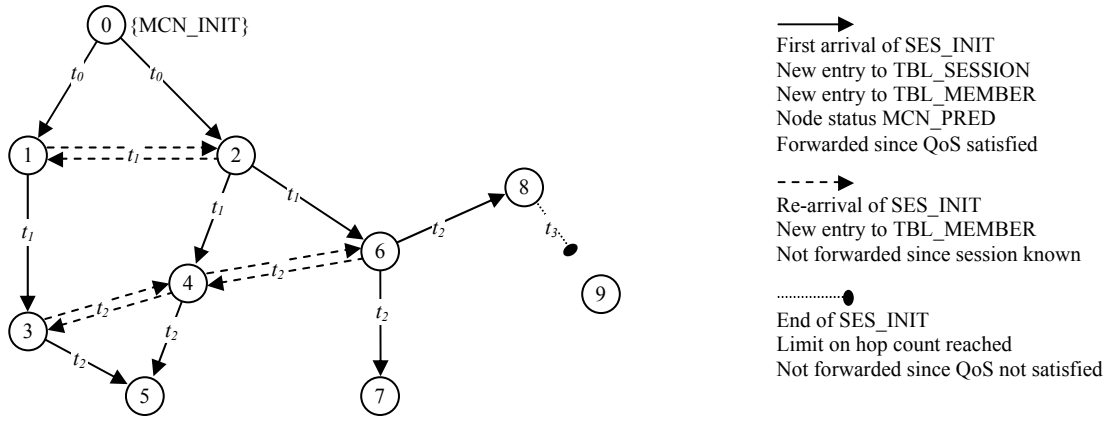


Fig. 1: The AQM session initiation process: SES_INIT is broadcast by MCN_INIT n_0 for a new session. It propagates through the network with time, informing all the nodes from n_1 to n_8, which update their TBL_SESSION and TBL_MEMBER. n_9 is not informed since it is beyond the QoS limits in terms of hop count. $t_i < t_{i+1}$, $0 \le i \le 3$, represent the relative timing of the messages. Upstream re-arrival arrows are not shown to keep the figure simple.

The session is closed, again by its initiator, with a session termination message (SES_TERMINATE). Upon receiving it, all nodes knowing that session clean their tables, whereas nodes forwarding multicast data also free their resources allocated to it. A node receiving a session termination packet forwards it if it has also forwarded the corresponding initiation packet or is currently forwarding session data to at least one active session member. Thus, receivers of a closed session are forced to leave the session.

3.3. Membership Management

Nodes can only join sessions known to them. A node can directly join a session if it is already a forwarding node in that session. Otherwise, it has to issue a join request. When a node broadcasts a join request packet (JOIN_REQ) for a session, only upstream neighbours which are aware of the session take it into consideration. The upstream flow of the request is guaranteed by comparing the hop count information of the packet with the distance to the server of the related session at each intermediate node. These nodes also maintain a temporary request table (TBL_REQUEST) to keep track of the requests and replies they have forwarded and prevent false or duplicate packet processing. The predecessors of the requester propagate the request upstream as long as QoS can be satisfied. Ad hoc networks are highly dynamic, and available resources may change considerably after the arrival of the initial QoS conditions with the session initiation packet. Therefore, QoS conditions are checked at each node to make sure that the current situation on resource availability allows the acceptance of a new session.

A forwarded request eventually reaches nodes that are already members of that session and therefore can directly send a reply (JOIN_REP) back to the requester. Members of a session are the initiator, forwarders, and receivers. Downstream nodes, having forwarded join requests and waiting for replies, aggregate the replies they receive at the end of a predefined amount of time and forward only the reply offering the best QoS conditions towards the requester. The information on the originator and the immediate forwarder of the reply is kept in the packet. The originator of the join request selects the one with the best QoS conditions among possibly several replies it receives. It changes its status from predecessor to receiver (MCN_RCV) and sends a reserve message (JOIN_RES) to the selected node which has forwarded the reply.

Upon receiving the reserve packet, intermediate nodes check whether they are among the intended forwarders on the path from the selected replier towards the requester. If this is the case, they change their status from predecessor to forwarder (MCN_FWD), reserve resources, and update their membership tables to keep a list of successors for that session. They forward the message upstream.

Eventually, the reserve message reaches the originator of the reply, which can be the session initiator with some or without any members, a forwarder with one or more successors, or a receiver. If the replier is the session initiator and this is its first member, it changes its status from initiator to server (MCN_SRV). If it is a receiver, it becomes a forwarder. In both cases, the replier records its successor in its member table and reserves resources to start sending multicast data. If the node is an active server or forwarder, it must have already reserved resources. It only adds the new member to its member table and continues sending the regular multicast data. Figure 2 shows the phases of joining a session.

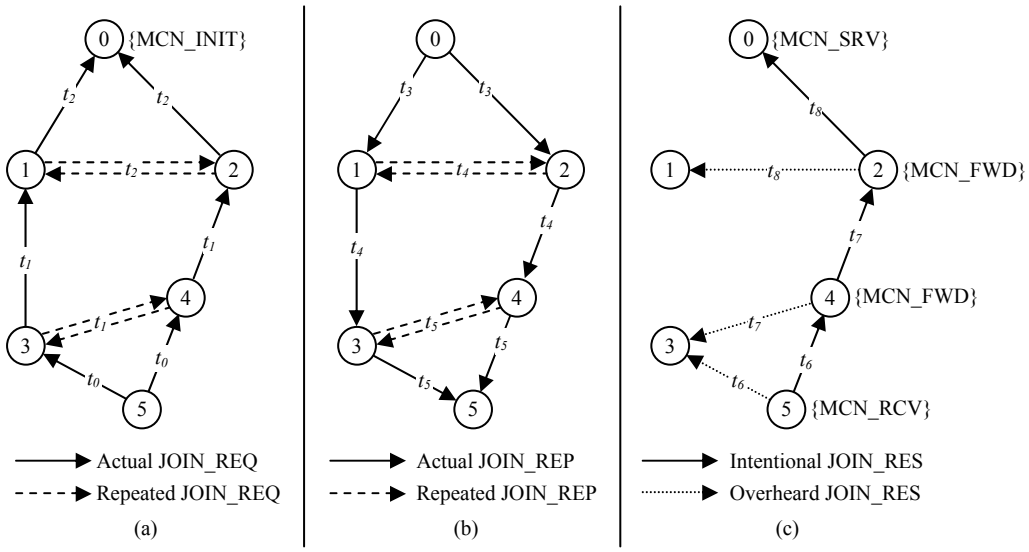


Fig. 2. The AQM session joining process: (a) JOIN_REQ is issued by n_5. It propagates towards any member of the session as long as QoS can be satisfied. Nodes from n_1 to n_4 update their TBL_REQUEST as they forward the packet since they are not session members. (b) JOIN_REP is sent back from n_0 to n_5. It is forwarded by n_1, n_2, n_3, n_4. (c) n_5 sends JOIN_RES along the selected QoS path via n_4, n_2, n_0, which reserve resources and update their status. Other nodes ignore the message. $t_i < t_{i+1}$, $0 \le i \le 8$, represent the relative timing of the messages. Upstream re-arrival arrows are not shown to keep the figure simple.

Intermediate nodes having sent a reply to a join request do not actually reserve resources until they receive the corresponding reservation packet. Since it is not certain that they are on the best QoS path according to the requester, they wait for this confirmation before activating that reservation and updating their resource availability data. Therefore, it is possible that a node receives simultaneous join requests for other sessions before having made that final reservation for an ongoing join process and replies the new requests although the cumulative QoS requirements cannot be satisfied anymore. If multiple requesters select the path via the common replier as their path to join their multicast sessions and send their reservation packets in that direction, the node grants only the reservation that arrives first and sends reservation error messages (JOIN_ERR) to the others to let them know that the final reservation of resources fails for them. In this case, the failing requesters select the next best replies from their request tables and try these alternatives by sending their reservation messages to the nodes which have sent them.

Each time a request-reply-reserve process completes successfully, intermediate nodes gather enough routing and membership data to take part in the packet forwarding task. When a host sends multicast packets with a particular multicast session ID, its neighbours already know if they are involved in the session by checking their tables, one with information on their own membership status, and another with a list of multicast sessions they are responsible of forwarding. Intermediate nodes also make use of the replies they receive during a session join process. If the reply is sent by a previously unknown predecessor in response to a request it has forwarded for a session, the intermediate node enters that predecessor into its member table for possible future routing operations.

A node needs to inform its forwarder on the multicast graph upon leaving a session. After receiving a quit notification (SES_LEAVE), the forwarding node deletes the leaving member from its member table. If this has been its only successor in that session, the forwarding node checks its own status regarding the session. If the forwarder itself is also a receiver, it updates its status. Otherwise, it frees resources and notifies its predecessor of its own leave.

3.4. Resource Allocation

The nodes in an ad hoc network have to maintain their resource information with as much accuracy as possible, which includes the ability to keep track of available bandwidth within their transmission range, in order to provide their neighbours with valid QoS routes when asked to take part in a request-reply-reserve process of a node wishing to join a multicast session. In AQM, nodes behave proactively with regard to multicast session information management by maintaining routing tables. Via periodic session updates, they keep themselves and their neighbours aware of the changes in the QoS conditions and node connectivity regarding the multicast sessions known to them. The rationale behind this behaviour is that QoS management in a highly dynamic environment such as wireless mobile networks cannot be achieved satisfactorily without informing the network on these issues in advance. However, the nature of a join process is on-demand, and AQM also checks the most up-to-date QoS conditions during this process and thus, it presents a hybrid approach.

The streaming nature of multimedia applications necessitates a pipelined approach to checking bandwidth availability. Concerning a session server about to allocate resources for its first member, twice as much bandwidth has to be available in the neighbourhood than the amount required by the QoS class of the session. The forwarding node immediately following the server on the path to the member belongs to the same neighbourhood as the server and shares the same free bandwidth. Therefore, a session server has to ensure that its successor also has enough bandwidth available to forward multicast data packets that it receives. On the other hand, a forwarder has to deal with its predecessor as well as its successor. Once the multicast session starts, it receives packets from its predecessor, rebroadcasts them, and allows its successor to forward the packets further downstream. Therefore, an intermediate node about to take part in the packet forwarding process has to check for availability of three times as much bandwidth than the amount needed by the session, since it shares the available bandwidth of the same neighbourhood as its immediate predecessor as well as its immediate successor. Thus, nodes have to check for availability of the necessary bandwidth according to their position within the multicast tree before accepting a request. When it is time to allocate resources, however, each node is responsible only for itself, i.e., nodes allocate only the amount of bandwidth that is necessary for the session of a particular QoS class. Figure 3 shows the pipelined approach to checking bandwidth availability.

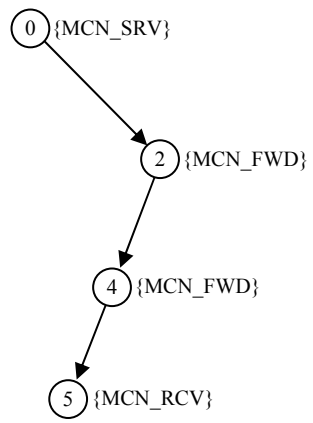


Fig. 3. The pipelined approach to checking bandwidth availability. n_0 checks for two times QoS bandwidth since it has to make sure that n_2 can also forward packets. n_2 checks for three times QoS bandwidth since, in addition to its predecessor n_0 and itself, it has to make sure that n_4 can also forward the streaming data. Finally, n_4 checks again for two times QoS bandwidth since n_5 is only a receiver which does not send packets.

Due to the broadcasting nature of the wireless medium, residual capacities are node-based, i.e., a node's available bandwidth is the residual capacity in its neighbourhood. In an ideal model, it is assumed that the bandwidth of a link can be determined on its neighbouring links [23]. Thus, each node calculates its current bandwidth usage by aggregating the bandwidth requirements of the multicast sessions it takes part in as a server or forwarder as follows:

$$B_{i,used} = \sum_{i \in \{m_s, m_f\}_j^S} B_j^S \quad (1)$$

where S stands for session, m_s is a serving member and m_f represents a forwarding member.

After making this calculation, each node informs its neighbours on the total amount of bandwidth it allocates to ongoing sessions via periodic greeting messages, which are introduced in the next section. The used bandwidth in a neighbourhood is the sum of the capacities allocated by all the nodes in that neighbourhood. Being informed on the bandwidth usage of all its neighbours, a node calculates the total bandwidth allocation in its neighbourhood as follows:

$$B_{i,used}^N = \sum_{k \in \{n\}_i^N} B_{k,used} \quad (2)$$

where N stands for neighbourhood and n is a neighbour.

Thus, each node can derive the remaining capacity available in its neighbourhood by subtracting this amount from the maximum capacity provided by the lower layers and the wireless medium:

$$B_{i,free}^N = B^{Max} - B_{i,used}^N \quad (3)$$

This is the maximum amount of free bandwidth the node can allocate to new session join requests:

$$B_{i,free} = B_{i,free}^N \quad (4)$$

It is noteworthy that for each node, (1) changes whenever the node gets an active role in a new session or is released by an ongoing session, whereas (2) can change with every greeting message from a new or existing neighbour. Thus, (1), (2), (3) and (4) have to be updated each time the node is about to decide to accept or reject a new session join request. On the other hand, these calculations are not necessary if the request is for a session that the node is already serving, which means that it has already allocated the necessary resources.

3.5. Neighbourhood Maintenance

Each node periodically broadcasts greeting messages (NBR_HELLO), informing its neighbours on its existence as well as its bandwidth usage, which is determined by the QoS classes of the sessions being served or forwarded by that node. Greeting messages can be piggybacked to other control and data messages to reduce control overhead. In other words, nodes do not need to send greeting messages explicitly unless they have not sent any piggybacked greeting messages for a certain period of time. Each node aggregates the information it receives with these messages in its neighbourhood table (TBL_NEIGHBOUR). This table is used to calculate the total bandwidth currently allocated to multicast sessions in the neighbourhood. Neighbourhood tables also help nodes with their decisions on packet forwarding. Session initiation packets are forwarded only if a node has neighbours other than its predecessors for that session.

In order to maintain connectivity and support QoS with maximum possible accuracy and minimum overhead under mobility conditions within their neighbourhood, nodes perform periodic cleanup operations on their session, membership and neighbourhood tables. If a node does not receive any greeting messages from a neighbour for a while, it considers that neighbour lost. Lost neighbours are marked as such for a predefined short period of time, at the end of which they are deleted from the neighbourhood table if they do not reappear. To prevent unnecessary message exchanges, nodes need to detect new neighbours quickly and distinguish them from lost neighbours reappearing after a short period of time and do not necessitate any update. If the lost neighbour is related to a session, it is also removed from the session, membership and request tables. This is an essential operation to keep the nodes up-to-date regarding the sessions and ready for future membership management activities such as initiating a new join request or replying to other nodes' join requests.

Additional action can be necessary depending on the status of the lost neighbour as well as that of the node itself. When an active session member, e.g., a forwarder or a receiver, loses its own forwarder or server, this means that it loses its connection to the session. It changes its status to a predecessor if there are other predecessors recorded in its membership table for this session. It also informs its successors with a lost session message (SES_LOST) if it is a forwarding member of the session. When, on the other hand, a server or a forwarder loses a receiver, it updates its status depending on the existence of other receivers in that session. It does not need to inform other nodes. When a node loses its only predecessor for a specific session due to changes in network topology, it notifies its successors of the lost session and lets them know that it should be deleted from the list of predecessors for that session. Downstream nodes receiving the lost session messages interpret them in a similar way to update their status regarding the lost session and forward the message if necessary. This mechanism, combined with the periodic updates mentioned previously, keeps nodes up-to-date regarding the QoS status of the sessions and prevents them from making infeasible join requests in terms of resource allocation.

4. Performance Evaluation

Previous research efforts have essentially been evaluated through the use of several important metrics which give a notion about the internal efficiency of the developed protocol. These are data delivery ratio in terms of data bytes or packets sent, and control overhead in terms of control bytes or packets sent, all measured per data byte or packet delivered [26]. However, the evaluation of QoS performance in ad hoc networks necessitates additional criteria. The main concern of this work is to evaluate the efficiency of AQM in providing multicast users with QoS and satisfying application requirements. Therefore, two new performance metrics, member and session satisfaction grades are introduced in the following sections.

The simulations are conducted using OPNET Modeler 10.5 Educational Version and Wireless Module [27]. AQM nodes are modelled in three layers with application, session, and network managers. The application manager is responsible for selecting the type of application to run, setting its QoS requirements, as well as making decisions on session initiation, termination, join and leave. The session manager is responsible for declaring new sessions initiated by its application manager to other nodes, sending requests for sessions its application manager wishes to join, keeping lists of sessions, members and requests of other nodes, processing and forwarding their control messages, and taking part in their join processes when necessary. The network manager is responsible for packet arrival and delivery, in addition to broadcasting periodic greeting messages and receiving other nodes' greeting messages in order to process them to derive free bandwidth information.

The simulations are repeated multiple times for each of the ad hoc network variables being experimented upon. The results are aggregated for a multicasting scenario with four QoS classes to represent a sample set of applications. Sessions are assigned randomly to one of these four classes defined in Table 1. The effect of mobility on the performance of AQM is observed under the random waypoint mobility model with various degrees of node mobility generated by uniformly distributed node speeds and pause times. In contrast to previous performance evaluations in the research literature, which limit their simulations to a few minutes, four hours of network lifetime have been simulated to get a realistic impression of the aggregated behaviour of multiple multicast sessions being maintained simultaneously in a distributed manner. The simulation parameters are given in Table 2.

Table 1
QoS Classes and Requirements

QoS Class	Bandwidth Requirement	Average Duration	Delay Tolerance	Application Type
0	128 Kbps	1 200 s	10 ms	High-quality voice
1	256 Kbps	2 400 s	90 ms	CD-quality audio
2	2 Mbps	1 200 s	10 ms	Video conference
3	3 Mbps	4 800 s	90 ms	High-quality video

Table 2
Simulation Parameters

Parameter Description	Value
Area size	1 000 m x 1 000 m
Greeting message interval	10 s
Maximum available bandwidth	10 Mbps
Mobility model	Random waypoint
Node distribution (initial)	Uniform
Node idle time	300 s (exponential)
Node pause time	40-400 s (uniform)
Node speed	1-4 m/s (uniform)
Service class distribution (multiclass)	0:40%; 1:20%; 2:30%; 3:10%
Session generation / joining ratio	1 / 9
Simulation duration	4 h
Wireless transmission range	250 m

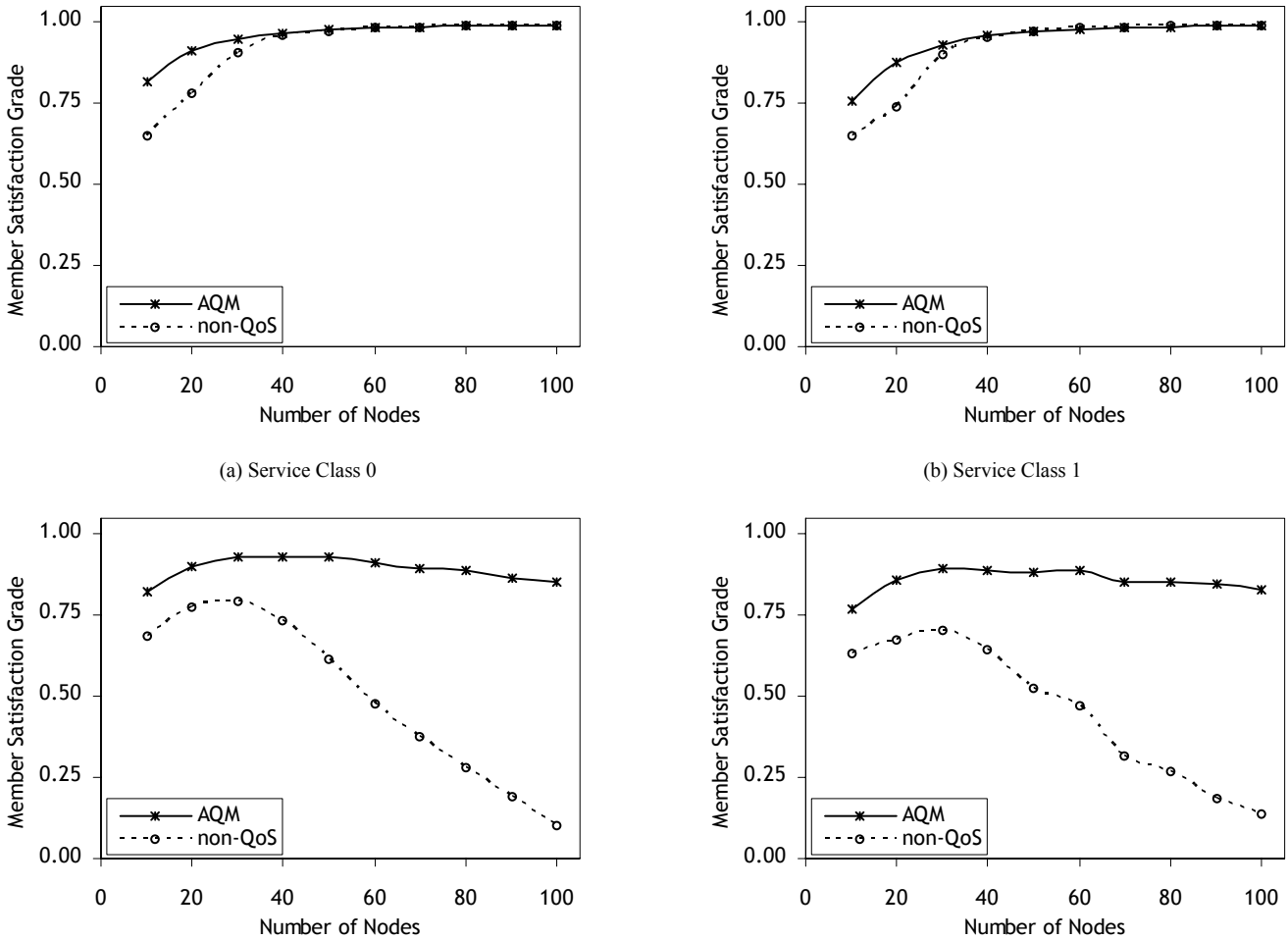
A node can take part in only one application at a time as a server or receiver. However, it can participate in any number of sessions as a forwarder as long as QoS conditions allow. The usage scenarios consist of open-air occasions such as search and rescue efforts and visits to nature in an area with boundaries, where a network infrastructure is not available and nodes move around with walking or running speeds.

4.1. Member Satisfaction Grade

An important aspect of the QoS-related multicasting decisions made by AQM is the improvement in the ratio of overloaded member nodes, which has a direct impact on the satisfaction of session members regarding the multicasting service provided. On the other hand, the same decisions lead the network to reject more join requests than a non-QoS scheme. Thus, the member satisfaction grade S_{Member} is defined as the weighted sum of these two components to evaluate the member-level success ratio of AQM, and formulated as follows:

$$S_{Member} = \beta\left(1 - \frac{o}{s + \alpha f}\right) + (1-\beta)\frac{r}{q} \qquad (5)$$

where o represents the number of overloaded nodes, which have decided to serve and forward more sessions than is possible without exceeding the maximum available bandwidth. s is the total number of session servers, and f is the total number of session forwarders. As mentioned in Section 3.4, the continuous flow of data in multimedia applications and the broadcasting nature of the wireless medium cause servers and forwarders to have different bandwidth considerations in their neighbourhoods. Thus, the impact of overloaded neighbours on these nodes is not the same. To reflect this difference, f is multiplied by a coefficient α, where $\alpha = 0.5$ in the simulations. The division $o/(s+\alpha f)$ gives the ratio of overloaded nodes to all serving and forwarding nodes. Thus, the first term of the summation, multiplied by a relative weight coefficient β, represents a member overload prevention rate. Continuing with the second term, r is the number of receivers, and q is the total number of join requests issued by all mobile nodes. Their ratio reflects the success of the scheme in



(a) Service Class 0

(b) Service Class 1

(c) Service Class 2

(d) Service Class 3

Fig. 4. Comparison of the member satisfaction grades of AQM to a non-QoS scheme for individual service classes. ($\alpha = 0.5$; $\beta = 0.5$)

satisfying a node's request to join a session. A success rate similar to the second term has previously been defined with the number of successful QoS multicast routes divided by the total number of requests [19]. The purpose of β, where $0 \leq \beta \leq 1$, is to adjust the relative weight of the member overload prevention rate over the member acceptance ratio according to the preferences of the ad hoc network.

Figure 4 compares AQM to the non-QoS scheme with regard to the member satisfaction grades for all service classes separately. For this set of simulations, $\beta = 0.5$ in order to observe the overload prevention and member acceptance abilities of the two competing schemes with equal priority. Other values are possible to change the network preferences and observe the effect of each component in the formula.

In Fig. 4(a) and (b), it can be seen that the member satisfaction grades are relatively low for sparse networks due to low connectivity and thus, more dynamic changes in network topology. Since service classes 0 and 1 do not require high amounts of bandwidth, no significant overload is caused. Therefore, the performance comparison is mainly based on the member acceptance ratios of the two schemes. AQM can cope with the network conditions more efficiently by informing the nodes on lost sessions and preventing infeasible join requests, whereas the non-QoS scheme generates requests to join sessions which are not reachable any more and causes more rejections. As the network density increases, the low connectivity problem diminishes and the performances of the schemes converge.

In Fig 4(c) and (d), a similar trend is observed under low connectivity. However, service classes 2 and 3 are more resource-demanding applications than classes 1 and 2. Therefore, once the network becomes better connected, the performance of the non-QoS scheme degrades rapidly since its nodes start making infeasible requests and become easily overloaded. AQM, on the other hand, maintains its member satisfaction by keeping its nodes informed on the sessions it can allow them to join. Thus, AQM performs significantly better than the non-QoS scheme with increasing node density, especially for applications of higher QoS classes.

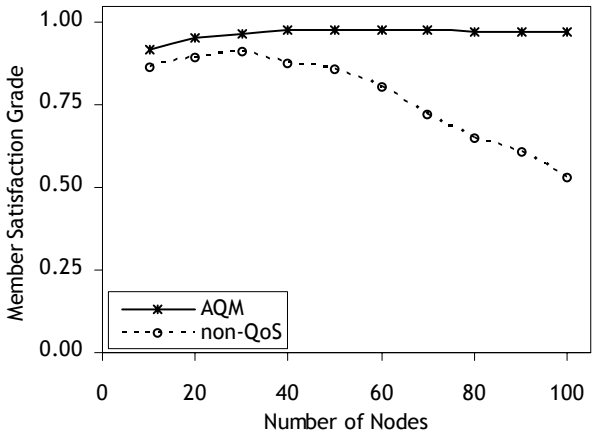


Fig. 5. Comparison of the member satisfaction grade of AQM to a non-QoS scheme as a function of increasing network population under the support for multiple service classes. ($\alpha = 0.5$; $\beta = 0.8$)

Figure 5 compares the member satisfaction grades of AQM to a non-QoS scheme under the support for multiple simultaneous service classes. Similar to the single-class cases, especially classes 2 and 3 presented in Fig. 4(c) and (d), respectively, AQM presents better results than the non-QoS scheme and the difference in the member satisfaction grades becomes clearer. It can be concluded that AQM tends to achieve more stable member satisfaction grades as the network population increases.

In AQM, where QoS support is active, nodes do not accept more traffic than the bandwidth available in their neighbourhood. However, overloaded members still exist due to a particular occurrence of the hidden terminal problem. Although none of these nodes allocates more bandwidth than available within its neighbourhood, this problem is a result of the allocations made by its neighbours, which cannot directly detect each other in the wireless medium, where resources are shared. In other words, a node can be surrounded by several neighbours, some of which are not within the transmission range of each other. Looking from the viewpoint of the node in the centre of that neighbourhood, it is possible that two of its neighbours not aware of each other can allocate resources in such a way that neither of them violates the QoS requirements within its own neighbourhood, whereas their total allocation exceeds the capacity limit of their common neighbour. In this case, the node in the centre experiences overload due to excessive resource usage in its neighbourhood, which cannot be prevented since its surrounding nodes cannot be informed about each other's reservations. Thus, the hidden terminal problem prevents nodes from making more accurate reservation decisions.

When QoS support is deactivated, nodes do not check their bandwidth before replying join requests. Thus, none of the requests are rejected. However, more serving and forwarding nodes become overloaded than AQM, which affects all their successors in all the sessions they serve. They start suffering from collisions and packet losses. As the number of nodes grows, more sessions are initiated, and more requests are accepted per node without taking care of the available bandwidth, which causes a drastic decrease in member satisfaction grades of the non-QoS network.

Figures 4 and 5 show that the performance of AQM is significantly better than the non-QoS scheme. QoS support increases member satisfaction grades during multicast sessions, especially as the number of nodes increases. While the application of QoS restrictions causes more users to be rejected, lack of these restrictions yield to catastrophic results. Without a policy to manage network resources effectively, users experience difficulties in getting any service as the network population grows and bandwidth requirements increase.

4.2. Session Satisfaction Grade

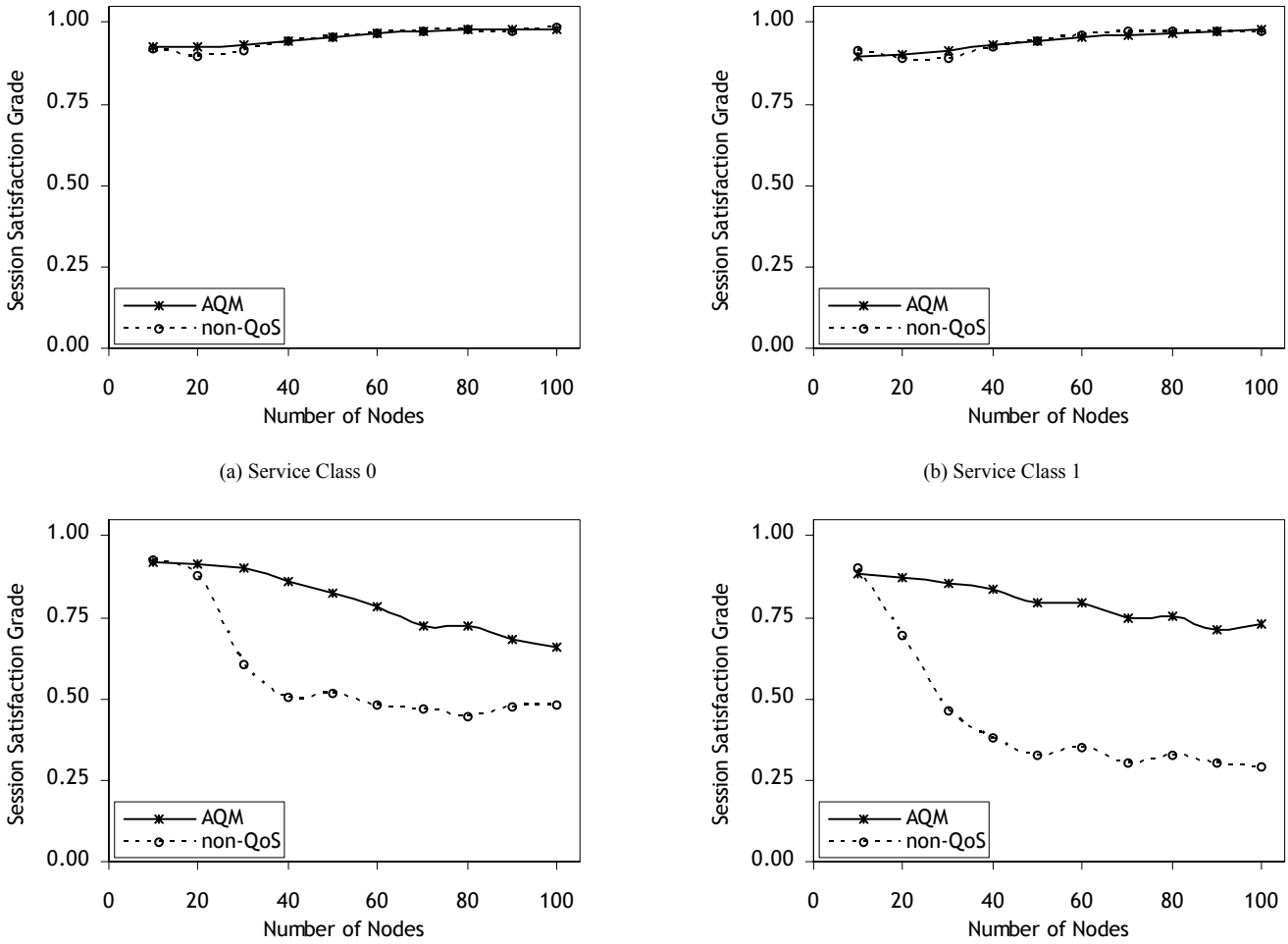
Rejection of some join requests and excessive bandwidth occupation by single nodes during the course of a session has also an effect on other members of that session. Therefore, it is necessary to observe the implications of these events on sessions as well. Thus, the session satisfaction grade $S_{Session}$ is defined as the weighted sum of these two components to evaluate the session-level success ratio of AQM, and formulated as follows:

$$S_{Session} = \gamma\left(1-\frac{l}{m}\right)+(1-\gamma)\left(1-\frac{j}{m}\right) \qquad (6)$$

where l is the number of sessions with at least one overloaded member, j is the number of sessions with at least one rejected join request, and m is the total number of sessions. The first term is the ratio of sessions without any overloaded members, which can be interpreted as a session-level overload prevention factor, whereas the second term reflects the success of AQM with regard to sessions without any rejections. The purpose of γ, where $0 \le \gamma \le 1$, is to adjust the relative weight of the overload prevention ratio over the rejection avoidance ratio according to the preferences of the network.

Figure 6 compares the session satisfaction grades of AQM to the non-QoS scheme for all service classes separately in networks of different sizes. For this set of simulations, γ = 0.8 in order to explicitly stress the effect of overloaded sessions on the multicast network performance. Other values are possible to increase the weight of the sessions with rejected join requests and observe their effects.

In Fig. 6(a) and (b), the session satisfaction grades are relatively low for networks with small numbers of nodes due to low connectivity. Similar to the observations made on member satisfaction, service classes 0 and 1 do not cause significant overload. Thus, the results mainly reflect the performances of the two schemes with regard to their ratios of sessions with rejected join requests. AQM performs slightly better in sparse networks since it informs nodes on lost sessions and prevents infeasible join requests.



(a) Service Class 0

(b) Service Class 1

(c) Service Class 2

(d) Service Class 3

Fig. 6. Comparison of the session satisfaction grades of AQM to a non-QoS scheme for individual service classes. ($\gamma = 0.8$)

As the network population increases, however, the two schemes start performing similarly.

In Fig 6(c) and (d), where service classes 2 and 3 have higher QoS demands, it can be seen that AQM performs significantly better than the non-QoS scheme. AQM keeps its session satisfaction grade at a higher level than the non-QoS scheme by preliminarily eliminating infeasible join requests as much as possible and also by rejecting them when they occur despite the countermeasures it takes. Thus, it tries not to allow session members to become overloaded and to lower its rejection rate at the same time. Overloaded sessions still exist since some of their members become overloaded as a result of the hidden terminal problem mentioned previously. However, its effect on the session satisfaction is limited. On the other hand, the non-QoS lets its nodes make infeasible requests and become overloaded, although its rejection rates are lower than AQM.

Figure 7 compares the session satisfaction grades of AQM to the non-QoS scheme under the support for multiple simultaneous service classes. As in the single-class case, AQM presents better results than the non-QoS scheme. Moreover, AQM performance degrades gracefully as the network population grows. It can be concluded that AQM also tends to achieve more stable session satisfaction grades as the network population increases.

AQM rejects some of the join requests to prevent existing session members from being overloaded, and provides better conditions for them. Thus, QoS support protects more sessions from being overloaded with an acceptable number of sessions with rejected members. Consequently, AQM achieves significant improvements in session satisfaction. It can be argued that AQM favours the satisfaction of sessions over the satisfaction of individual members. In the non-QoS scheme, nodes directly accept join requests as soon as they are aware of any path towards the session server. Since they are not limited by their resource availability, they cause more session members to be overloaded, resulting in decreased session satisfaction grades. Due to low connectivity, there is not a significant difference between the results for small networks. As the

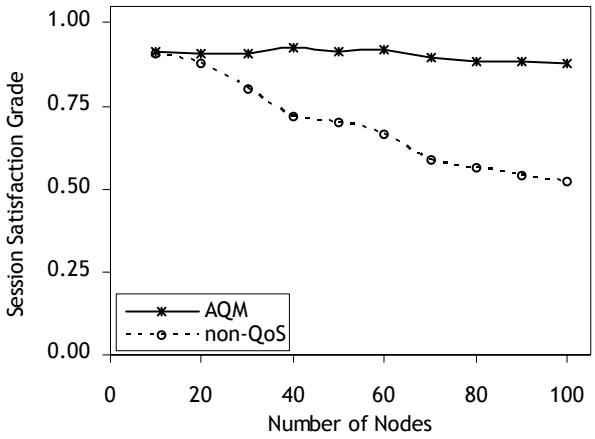


Fig. 7. Comparison of the session satisfaction grade of AQM to a non-QoS scheme as a function of increasing network population under the support for multiple service classes. ($\gamma = 0.8$)

network size grows, however, the lack of bandwidth restrictions causes more nodes to become overloaded. More sessions are unsatisfied, and session satisfaction degrades.

Figures 6 and 7 show that QoS support also increases session satisfaction grades significantly. The degradation in session satisfaction as network sizes grow and bandwidth requirements increase is clearly less dramatic for AQM than a non-QoS scheme. It can be said that an overloaded member triggers decreased performance on all the sessions it has been serving when the overload occurs, whereas the rejection of a single user has a limited effect on the session satisfaction. Thus, AQM achieves better performance by decreasing the number of overloaded members and sessions while keeping the number of rejected join requests at an acceptable level.

5. Conclusion

Expectations of wireless users shifting towards high quality, group-oriented, mobile multimedia communication affects the needs of today's networks. The QoS scheme presented in this work, AQM, provides ad hoc networks with QoS support and multicasting features. It improves multicasting efficiency through resource management. AQM checks bandwidth availability within each node's neighbourhood based on previous reservations, and ensures that updated QoS information is used to select routes that can meet service requirements of a session.

The primary evaluation criteria for AQM are service satisfaction ratios defined both at member and session levels. AQM is compared to a non-QoS scheme with regard to these criteria. Simulations show that there is a significant performance difference between the two schemes at both levels. By applying QoS restrictions to the ad hoc network, AQM achieves better satisfaction grades and improves the multicasting efficiency for members and sessions. Without a QoS scheme, users experience difficulties in getting the service they demand as the network population grows and bandwidth requirements increase. AQM proves that QoS is essential for and applicable to ad hoc multimedia networks.

The performance criteria proposed in this article can be further developed by adding delay-based components to the formulae for member and session satisfaction. Another improvement is a metric to measure the session losses experienced and rejoin attempts made by members. Some of the recent multicast routing protocols in the literature can be assessed using these criteria to have an alternate view to their performance in terms of QoS as experienced by the user. Another interesting future work is the development of an adaptive system which tries to maximise an objective function based on these criteria by applying alternate AQM strategies in accordance with the coefficients given to the mobile nodes when they enter the AQM domain.

AQM has a simple and flat network structure where all nodes are equal. It avoids complicated network topologies such as hierarchical or clustered structures, which are challenging in terms of design and maintenance and present points of failure. However, it is possible to adapt AQM to a clustered network to scale with network size. Intra-cluster multicast sessions can be handled by AQM, whereas inter-cluster communication can be managed by a higher-layer, hierarchical version of the same protocol, still providing the network with QoS features. It is not a realistic assumption that a mobile network can afford a pure on-demand scheme if it has to support QoS. Therefore, AQM proposes a hybrid method in terms of multicast routing with table-driven session management and on-demand verification of QoS information upon the initialisation of a join process.

An important future research direction for AQM, which is closely related to QoS data accuracy, is the development of a solution to the hidden terminal problem. The approach to residual bandwidth calculation presented in this work provides a sufficient method to measure bandwidth availability within a neighbourhood. However, it does not consider bandwidth usage beyond direct neighbours. Thus, it is susceptible to hidden terminal problems and therefore needs further research. To overcome this problem, an extension to the request-reply-reserve process is necessary, whereby each replying node consults its neighbourhood to see if there are any objections.

Another point to consider for future research is the observation of the effects of the underlying data link layer protocols on the performance of AQM and the development of necessary means to improve this performance where necessary. Neighbour awareness and discovery are typically left to the MAC protocol and handled by a periodic mechanism of beacons. However, reliable MAC broadcast is a challenging task due to the request-to-send/clear-to-send (RTS/CTS) signalling problem. The MAC layer is also responsible for resource reservation and the acquisition of available link bandwidth information, which is another significant task since it involves

infrastructure decisions. However, AQM is independent of the design of lower layers, and within the scope of this work, efforts have been made to maintain its integrity by addressing these issues in higher-layers.

References

[1] Obraczka, K, and G. Tsudik, "Multicast Routing Issues in Ad Hoc Networks," *Proceedings of IEEE ICUPC,* Florence, Italy, October 1998.

[2] Van Nee, R, and R. Prasad, *OFDM for Wireless Multimedia Communications*, 1st edition, Artech House Publishers, 2000.

[3] Chakrabarti, S, and A. Mishra, "QoS Issues in Ad Hoc Wireless Networks," *IEEE Communications Magazine*, Vol. 39, No. 2, pp. 142-148, February 2001.

[4] Zhang, Q, W. Zhu, G.J. Wang, and Y.Q. Zhang, "Resource Allocation with Adaptive QoS for Multimedia Transmission over W-CDMA Channels," *Proceedings of IEEE WCNC,* Chicago, USA, September 2000.

[5] Walrand, J, and P. Varaiya, *High-Performance Communication Networks*, 2nd edition, Morgan Kaufmann Publishers, 2000.

[6] Royer, E.M, and C.E. Perkins, "Multicast Operation of the Ad-Hoc On-Demand Distance Vector Routing Protocol," *Proceedings of ACM/IEEE MOBICOM,* Seattle, USA, August 1999.

[7] Royer, E.M, and C.E. Perkins, "Multicast Ad Hoc On-Demand Distance Vector (MAODV) Routing," *IETF MANET Working Group Internet Draft* , work in progress, July 2000.

[8] Garcia-Luna-Aceves, J.J, and E.L. Madruga, "The Core-Assisted Mesh Protocol," *IEEE Journal of Selected Areas in Communications,* Vol. 17, No.8, pp. 1380-1394, August 1999.

[9] Madruga, E.L, and J.J. Garcia-Luna-Aceves, "Scalable Multicasting: The Core-Assisted Mesh Protocol," *Kluwer Mobile Networks and Applications*, Vol. 6, No. 2, pp. 151-165, March 2001.

[10] Ozaki, T, J.B. Kim, and T. Suda, "Bandwidth-Efficient Multicast Routing for Multihop, Ad Hoc Wireless Networks," *Proceedings of IEEE INFOCOM,* Alaska, USA, April 2001.

[11] Bae, H.B, S.J. Lee, W. Su, and M. Gerla, "The Design, Implementation, and Performance Evaluation of the On-Demand Multicast Routing Protocol in Multihop Wireless Networks," *IEEE Network,* Vol. 14, No. 1, pp. 70-77, January 2000.

[12] Lee, S.J, W. Su, and M. Gerla, "On-Demand Multicast Routing Protocol in Multihop Wireless Mobile Networks," *ACM Mobile Networks and Applications*, Vol. 7, No. 6, pp. 441-453, December 2002.

[13] Lee, S, and C. Kim, "A new wireless ad hoc multicast routing protocol," *Elsevier Science Computer Networks*, Vol. 38, pp. 121-135, February 2002.

[14] Toh, C.K, Ad Hoc Mobile Wireless Networks: Protocols and Systems, 1st edition, Prentice Hall PTR, 2002.

[15] Ji, L, and M.S. Corson, "Differential Destination Multicast – A MANET Multicast Routing Protocol for Small Groups," *Proceedings of IEEE INFOCOM,* Alaska, USA, April 2001.

[16] Ji, L, and M.S. Corson, "Explicit Multicasting for Mobile Ad Hoc Networks," *ACM Mobile Networks and Applications*, Vol. 8, No. 5, pp. 535-549, October 2003.

[17] Sajama, S, and Z.J. Haas, "Independent-Tree Ad Hoc Multicast Routing (ITAMAR)," *ACM Mobile Networks and Applications*, Vol. 8, No. 5, pp. 551-566, October 2003.

[18] Chen, Y.S, Y.W. Ko, and T.L. Lin, "A Lantern-Tree-Based QoS Multicast Protocol for Wireless Ad Hoc Networks," *Proceedings of IEEE ICCCN,* Miami, USA, October 2002.

[19] Chen, Y.S, and Y.W. Ko, "A Lantern-Tree-Based QoS On-Demand Multicast Protocol for a Wireless Mobile Ad Hoc Network," *IEICE Transactions on Communications,* Vol. E87-B, No. 3, March 2004.

[20] Pompili, D, and M. Vittucci, "A Probabilistic Predictive Multicast Algorithm in Ad Hoc Networks (PPMA)," *Proceedings of MED-HOC-NET,* Mugla, Turkey, June 2004.

[21] Gerla, M, and J.T. Tsai, "Multicluster, Mobile, Multimedia Radio Network," *ACM/Baltzer Journal of Wireless Networks*, Vol. 1, No. 3, pp. 255-265, 1995.

[22] Zhu, C, and M.S. Corson, "QoS Routing for Mobile Ad Hoc Networks," *Proceedings of IEEE INFOCOM,* New York, USA, June 2002.

[23] Chen, S, and K. Nahrstedt, "Distributed Quality of Service Routing in Ad Hoc Networks," *IEEE Journal on Selected Areas in Communications*, Vol. 17, No. 8, pp. 1488-1505, August 1999.

[24] Chen, T.W, J.T. Tsai, and M. Gerla, "QoS Routing in Multihop, Multimedia, Wireless Networks," *Proceedings of IEEE ICUPC,* San Diego, USA, October 1997.

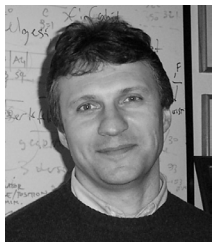
[25] Lin, C.R, and J.S. Liu, "QoS Routing in Ad Hoc Wireless Networks," *IEEE Journal on Selected Areas in Communications*, Vol. 17, No. 8, pp. 1426-1438, August 1999.

[26] Corson, S, and J. Macker, "Mobile Ad Hoc Networking (MANET): Routing Protocol Performance Issues and Evaluation Considerations," *RFC 2501*, January 1999.

[27] OPNET Technologies Inc, Bethesda, MD, USA, available at http://www.opnet.com

Vitae

Kaan Bür received his BS degree in control and computer engineering from Istanbul Technical University in 1995 and his MS degree in computer engineering from Bogaziçi University in 1998. He worked first as an I.T. engineer and then as an R&D engineer in Beko Elektronik A.S. between 1995 and 2002. Currently, he is a PhD candidate in the Computer Engineering Department of Bogaziçi University. His research interests include high-speed networks, wireless, mobile and multimedia communications. Kaan Bür is a student member of IEEE.



Cem Ersoy received his BS and MS degrees in electrical engineering from Bogaziçi University, Istanbul, in 1984 and 1986, respectively. He worked as an R&D engineer in NETAS A.S. between 1984 and 1986. He received his PhD in electrical engineering from Polytechnic University, Brooklyn, New York in 1992. Currently, he is a professor in and head of the Computer Engineering Department of Bogaziçi University. His research interests include performance evaluation and topological design of communication networks, wireless communications and mobile applications. Dr. Ersoy is a senior member of IEEE.