# Reliable Multicasting in MANETs

Thomas Kunz
Carleton University

Thomas Kunz
2 Beckington Private
Ottawa, Ont., Canada
K2P 2N5

Project Manager: Louise Lamont 991-9635

Contract Number: CRC 5004800

Contract Scientific Authority: Louise Lamont 991-9635

## Defence R&D Canada - Ottawa

Contract Report

DRDC-Ottawa

July 2003

## Communications Research Centre

## Abstract

Multicasting is the transmission of datagrams (packets) to a group of zero or more hosts identified by a single destination address. Maintaining group membership information and building an optimal multicast distribution structure (typically in the form of a routing tree) is challenging even in wired networks. However, nodes are increasingly mobile. One particularly challenging environment for multicast is a mobile ad-hoc network (MANET). This report provides an in-depth study of one-to-many and many-to-many communication in mobile ad-hoc networks. First we compare a range of best-effort protocols: 2 unicast routing protocols, 3 multicast routing protocols, and 2 broadcast protocols. Our results show that broadcast protocols, in particular BCAST, perform well and that this performance does not come with a high overhead. We then enhance BCAST with a NACK-based retransmission scheme to further increase the packet delivery ratio, resulting in reliable BCAST. We also explore the impact of the MAC layer on the performance of both best-effort BCAST and reliable BCAST. Varying the user traffic load and the MAC layer, the results provide a number of insights into the relationship between MAC and ROUTING layer. Overall, BCAST is a protocol that achieves high packet delivery, at the cost of an increase in packet latency. We show that the protocol performs well in a wide range of scenarios and over a number of MAC layers (all of which were variants of the 802.11 protocol family). Increasing packet delivery through a retransmission scheme is, however, only of limited value. As MAC rates increase for current and future networks, MANETs will be able to support a non-trivial amount of traffic per multicast sender. Achieving high packet delivery ratios in these networks can be achieved by adjusting the data volume through flow control to operate in the protocol “sweet spot”, using the best-effort protocol as basic protocol.



## Résumé

La multidiffusion consiste en la transmission de datagrammes (paquets) à un groupe qui compte aucun hôte ou plus et qui est identifié par une seule adresse de destination. La conservation des données sur les membres d’un groupe et la construction d’une structure de multidiffusion optimale (généralement en forme d’arborescence de routage) représentent un défi, même dans les réseaux câblés. Toutefois, les nœuds sont de plus en plus mobiles. Un réseau mobile ad hoc (MANET) est un environnement particulièrement complexe pour la multidiffusion. Le présent rapport propose une étude approfondie sur les communications de un à plusieurs ou de plusieurs à plusieurs dans les MANET. Cette étude compare tout d’abord une gamme des meilleurs protocoles : deux protocoles de routage à diffusion individuelle, trois protocoles de routage à diffusion sélective et deux protocoles à diffusion générale. Les résultats montrent que les protocoles à diffusion générale, BCAST en particulier, offrent un rendement intéressant et que ce rendement n’engendre pas de surcharge importante. Les chercheurs ont ensuite amélioré le protocole BCAST à l’aide d’un modèle de retransmission fondé sur des accusés de réception négatifs (NACK) afin d’accroître le rapport de remise des paquets, créant ainsi un protocole BCAST fiable. Le rapport examine aussi l’incidence d’une sous-couche de contrôle d’accès au support sur le rendement du meilleur protocole BCAST et du protocole BCAST fiable. Selon le volume du trafic des utilisateurs et la sous-couche de

contrôle d’accès au support, les résultats offrent un certain nombre d’indices sur la relation entre la sous-couche de contrôle d’accès au support et la couche de routage. En général, le protocole BCAST permet d’obtenir une remise importante des paquets au prix d’une augmentation du temps d’attente des paquets. Le rapport indique que le protocole s’exécute bien pour une vaste gamme de scénarios et un certain nombre de sous-couches de contrôle d’accès au support (qui sont toutes des variantes de la catégorie des protocoles 802.11). La valeur de l’augmentation de la remise des paquets à l’aide d’un modèle de retransmission est toutefois limitée. À mesure que les taux de contrôle d’accès au support augmenteront pour les réseaux actuels et futurs, les MANET pourront soutenir un volume considérable de trafic par expéditeur de multidiffusion. Il est possible d’obtenir des rapports élevés de remise de paquets dans ces réseaux en ajustant la quantité de données avec un contrôle de flux afin de demeurer à l’intérieur des paramètres idéaux du protocole et en utilisant le meilleur protocole comme protocole de base.

This page intentionally left blank.

# Executive summary

Multicasting is the transmission of datagrams (packets) to a group of zero or more hosts identified by a single destination address. A multicast packet is typically delivered to all members of its destination host group with the same reliability as regular unicast packets. In the case of IP Multicasting, for example, the packet is not guaranteed to arrive at all members of the destination group or in the same order relative to other packets.

Multicasting is intended for group-oriented computing and its use within a network has many benefits. Multicasting reduces the communication costs for applications that send the same data to multiple recipients. Instead of sending via multiple unicasts, multicasting minimizes the link bandwidth consumption, sender and router processing, and delivery delay. In addition, multicasting provides a simple yet robust communication mechanism whereby a receiver's individual address is unknown or changeable transpar ently to the source.

There are more and more applications where one-to-many or many-to-many dissemination is an essential task. The multicast service is critical in applications characterized by the close collaboration of teams (e.g. rescue patrol, battalion, scientists, etc) with requirements for audio and video conferencing and sharing of text and images. In the Internet (IPv4), multicasting facilities were introduced via the Multicast Backbone (MBone), a virtual overlay network on top of the Internet. This overlay network consists of multicast-capable islands connected by tunnels. Each island contains one or more special routers called multicast routers, which are logically connected by these tunnels. These routers manage group membership and cooperate to route data to all hosts wishing to participate in a multicast group. IP multicast groups are identified by special IP addresses. Support for multicasting is an integral component of IPv6, so it can be assumed that multicasting applications will become even more popular with the increased popularity and acceptance of IPv6.

Typically, the membership of a host group is dynamic; that is, hosts may join and leave groups any time. There is no restriction on the location or number of members in a host group. A host may be a member of more than one group at a time. A host does not have to be a member of a group to send packets to it. A host group may be permanent or transient. A permanent group has a well-known, administratively assigned address. It is the address, not the membership of the group that is permanent; at any time a permanent group may have any number of members, even zero. Those IP multicast addresses that are not reserved for permanent groups are available for dynamic assignment to transient groups which exist only as long as they have members.

Maintaining group membership information and building an optimal multicast distribution structure (typically in the form of a routing tree) is challenging even in wired networks. However, nodes are increasingly mobile. One particularly challenging environment for multicast is a mobile ad-hoc network (MANET). A MANET consists of a dynamic collection of nodes with sometimes rapidly changing multi-hop topologies that are composed of relatively low-bandwidth wireless links. There is no assumption of an underlying fixed infrastructure. Nodes are free to move arbitrarily. Since each node has a limited transmission range, not all messages may reach all the intended hosts. To provide communication through

the whole network, a source-to-destination path could pass through several intermediate neighbour nodes. Unlike typical wireline routing protocols, ad-hoc routing protocols must address a diverse range of issues. The network topology can change randomly and rapidly, at unpredictable times. Since wireless links generally have lower capacity, congestion is typically the norm rather than the exception. The majority of nodes will rely on batteries, thus routing protocols must limit the amount of control information that is passed between nodes. Also, multicast group members and other nodes move, thus precluding the use of a fixed multicast topology.

The goal of MANETs is to extend mobility into the realm of autonomous, mobile, wireless domains, where a set of nodes form the network routing infrastructure in an ad-hoc fashion. The majority of applications for the MANET technology are in areas where rapid deployment and dynamic reconfiguration are necessary and the wireline network is not available. These include military battlefields, emergency search and rescue sites, classrooms, and conventions where participants share information dynamically using their mobile devices. These applications lend themselves well to multicast operation. In addition, within a wireless medium, it is even more crucial to reduce the transmission overhead and power consumption. Multicasting can improve the efficiency of the wireless link when sending multiple copies of messages by exploiting the inherent broadcast property of wireless transmission. While many applications, such as audio/video distribution, can tolerate loss of data content, many other applications cannot. In addition, even loss-tolerant applications will suffer a performance penalty: an audio stream may experience a short gap or lower fidelity in the presence of loss.

This report provides an in-depth study of one-to-many and many-to-many communication in mobile ad-hoc networks. The original goal of this work was to design an efficient protocol that delivers packets from one or multiple senders to many receptions with high probability. We started this effort by exploring the performance of a number of best-effort protocols: 2 unicast routing protocols, 3 multicast routing protocols, and 2 broadcast protocols. The extensive simulation results in Section 4 show that broadcast protocols perform surprisingly well, and that this performance does not come with a high overhead. So we decided to use the more efficient broadcast protocol, BCAST, as starting point for the next step. After exploring a number of design alternatives (Section 5), we enhanced BCAST with a NACK-based retransmission scheme to further increase the packet delivery ratio. The simulation results reported in Section 6 demonstrate that this mechanism indeed increases the packet delivery ratio. The experiments also show that a high degree of mobility is actually advantageous: as network partitions are potentially short-lived, our retransmission scheme is more likely to successfully recover from packet losses during such partitions. In contrast, in networks with longer-lived partitions, the amount of packets buffered at nodes needs to be increased substantially to recover in these cases. Finally, the results show that implementing any reliability mechanism has to be done with care. As the network capacity is limited, flooding the network with NACKs and the ensuing packet re-transmission attempts will have a detrimental impact on the protocol performance when the network is experiencing congestion. In the reliable BCAST protocol, we therefore have each node monitor the local network traffic and suppress NACKs when it observed too much traffic in the recent past.

In a last step, we explored the impact of the MAC layer on the performance of both best-effort BCAST and reliable BCAST. Varying the user traffic load and the MAC layer, Section 7 discusses a number of insights into the relationship between MAC and ROUTING layer. In

particular, we noted that BCAST suffers as the number of MAC-level packet collisions increases. Therefore, approaches to reduce such collisions have the potential to increase the performance of BCAST substantially. The results also indicate that our NACK-based packet retransmission scheme increases packet delivery ratio in only a limited number of scenarios. If the user traffic, relative to the MAC layer data rate, is low, best-effort BCAST performs already extremely well. If, on the other hand, the network starts to experience congestion due to high user traffic, we have to throttle NACKs to prevent them from negatively impacting the protocol performance, so again there is little difference between the best-effort and reliable protocol versions. As the discussion in Section 7 also demonstrates, the NACK throttle has to be tuned carefully for the MAC layer, otherwise the resulting performance suffers.

In conclusion, BCAST is a protocol that achieves high packet delivery, at the cost of an increase in packet latency. We have shown that the protocol performs well in a wide range of scenarios and over a number of MAC layers (all of which were variants of the 802.11 protocol family). Increasing packet delivery through a retransmission scheme is, however, only of limited value. As MAC rates increase for current and future networks, MANETs will be able to support a non-trivial amount of traffic per multicast sender. Achieving high packet delivery ratios in these networks can be achieved by adjusting the data volume through flow control to operate in the protocol “sweet spot”, using the best-effort protocol as basic protocol.

.

# Sommaire

La multidiffusion consiste en la transmission de datagrammes (paquets) à un groupe qui compte aucun hôte ou plus et qui est identifié par une seule adresse de destination. La remise d'un paquet à diffusion sélective à tous les membres d'un g roupe de multidiffusion est généralement aussi fiable que celle d'un paquet à diffusion individuelle ordinaire. Dans le cas de la multidiffusion IP, par exemple, rien ne garantit la remise du paquet à tous les membres du groupe de destination ou le maintien de son ordre initial comparativement aux autres paquets.

La multidiffusion cible l'informatique axée sur les groupes, et son utilisation dans un réseau comporte de nombreux avantages. La multidiffusion réduit les coûts de communication des applications qui envoient les mêmes données à beaucoup de destinataires. Au lieu de recourir à l'envoi répété de diffusions ponctuelles, la multidiffusion minimise la consommation de largeur de bande de liaison, le traitement de l'expéditeur et du routage ainsi que le t emps de remise. En outre, la multidiffusion offre un mécanisme de communication simple mais robuste grâce auquel l'adresse personnelle du destinataire est inconnue de la source ou modifiable aisément.

Il existe de plus en plus d'applications pour lesquelle s la diffusion de un à plusieurs et de plusieurs à plusieurs est une fonction essentielle. Le service de multidiffusion est essentiel aux applications caractérisées par une étroite collaboration entre les équipes (p. ex. service d'urgence, bataillon, scien tifiques) et des exigences en matière de conférence vidéo et audio et de partage de textes et d'images. Dans Internet (protocole IPv4), les installations de multidiffusion ont été introduites au moyen du réseau Mbone, un réseau virtuel superposé à Internet. Ce réseau superposé est composé d'îlots fonctionnant en multidiffusion et reliés entre eux par des tunnels. Chaque îlot possède au moins un routeur spécial appelé « routeur de multidiffusion » qui est connecté de façon logique à ces tunnels. Ce routeur gère la composition des groupes et collabore afin de diriger les données à tous les hôtes qui souhaitent participer à un groupe de multidiffusion. Les groupes de multidiffusion IP sont identifiés par des adresses IP particulières. La prise en charge de la multidiffusion est une partie intégrante du protocole IPv6; il est donc permis de croire que les applications de multidiffusion seront de plus en plus utilisées avec l'adoption du protocole IPv6 et sa popularité accrue.

En général, l'adhésion à un groupe de multidiffusion est dynamique, ce qui signifie que les hôtes peuvent s'y joindre ou le quitter à tout moment. Il n'existe aucune limite concernant l'emplacement ou le nombre de membres d'un tel groupe. Un hôte peut être membre de plus d'un groupe à la fois et ne doit pas nécessairement faire partie d'un groupe pour lui envoyer des paquets. Un groupe de multidiffusion est temporaire ou permanent. Un groupe permanent possède une adresse bien connue et attribuée par l'administration. C'est l'adresse, et non l'adhésion au groupe, qui est permanente. À tout moment, un groupe permanent peut compter un nombre indéfini de membres; il peut même n'en avoir aucun. Les adresses de multidiffusion IP qui ne sont pas réservées pour des groupes permanents peuvent être attribuées de manière dynamique à des groupes temporaires qui existent aussi longtemps qu'ils ont des membres.

La conservation des données sur les membres d'un groupe et la construction d'une structure de multidiffusion optimale (généralement en forme d'arborescence de routage) représentent un défi, même dans les réseaux câblés. Toutefois, les nœuds sont de plus en plus mobiles. Un réseau mobile ad hoc (MANET) est un environnement particulièrement complexe pour la multidiffusion. Un MANET consiste en un regroupement dynamique de nœuds avec des topologies à plusieurs bonds qui changent parfois rapidement et qui sont composées de liens sans fil à débit relativement bas. Il n'y a aucune infrastructure fixe sous-jacente. Les nœuds peuvent se déplacer librement. Puisque chaque nœud possède une portée d'émission restreinte, il est possible que les destinataires visés ne reçoivent pas tous les messages. Pour transmettre les communications à l'ensemble du réseau, la voie de la source à la destination peut emprunter plusieurs nœuds intermédiaires voisins. Contrairement aux protocoles de routage câblé typiques, les protocoles de routage ad hoc doivent résoudre une vaste gamme de problèmes. La topologie du réseau peut changer rapidement et de manière aléatoire à n'importe quel moment. Puisque les liens sans fil ont généralement une capacité inférieure, la congestion est devenue la norme plutôt que l'exception. La majorité des nœuds sont alimentés par des piles; les protocoles de routage doivent donc limiter le volume d'information de contrôle transmis entre les nœuds. En outre, les membres du groupe de multidiffusion et les autres nœuds se déplacent, ce qui empêche l'utilisation d'une topologie de multidiffusion fixe.

L'objectif des MANET est d'étendre la mobilité aux domaines autonome, mobile et sans fil, où un ensemble de nœuds forment une infrastructure d'acheminement des données de manière ponctuelle. La plupart des applications destinées à la technologie des MANET sont dans des zones qui exigent une mise en place rapide et une reconfiguration dynamique et pour lesquelles aucun réseau sans fil n'est disponible. Cela inclut les champs de bataille, les sites de recherche et de sauvetage, les salles de classe et les congrès où les participants partagent des renseignements de façon dynamique à l'aide de leurs appareils mobiles. Ces applications conviennent parfaitement à la multidiffusion. En outre, dans un appareil sans fil, il est encore plus important de réduire la surcharge de transmission et la consommation d'énergie. La multidiffusion peut améliorer l'efficacité des liaisons sans fil lors de l'envoi d'un grand nombre de messages en exploitant la propriété inhérente de diffusion générale des transmissions sans fil. Bien que de nombreuses applications, comme la distribution audio et vidéo, puissent supporter une perte de contenu d'information, beaucoup d'autres applications ne peuvent tolérer cette situation. De plus, même les applications qui peuvent supporter de telles pertes verront leur rendement diminuer : le flot de données audio peut comporter des lacunes ou connaître une diminution de sa fidélité en présence de pertes.

Le présent rapport propose une étude approfondie sur les communications de un à plusieurs ou de plusieurs à plusieurs dans les MANET. L'objectif initial des travaux était de concevoir un protocole fiable qui distribue les paquets envoyés par un ou plusieurs expéditeurs à plusieurs destinataires avec un haut pourcentage de réussite. Cette étude compare tout d'abord une gamme des meilleurs protocoles : deux protocoles de routage à diffusion individuelle, trois protocoles de routage à diffusion sélective et deux protocoles à diffusion générale. Les résultats des nombreuses simulations de la section 4 montrent que les protocoles à diffusion générale offrent un rendement intéressant et que ce rendement n'engendre pas de surcharge importante. Les chercheurs ont donc décidé d'utiliser le protocole le plus fiable, le BCAST, comme point de départ de l'étape suivante. Après l'examen d'un certain nombre de conceptions possibles (section 5), ils ont amélioré le protocole BCAST à l'aide d'un modèle de retransmission fondé sur des accusés de réception négatifs (NACK) afin d'accroître le

rapport de remise des paquets. Les résultats de simulation rapportés dans la section 6 démontrent que ce mécanisme a effectivement augmenté ce rapport. Les expériences ont également établi qu'une mobilité accrue est un avantage : puisque les partitions du réseau sont potentiellement de courte durée, le modèle de retransmission est plus susceptible de mieux se remettre des pertes de paquets survenues durant ces partitions. En revanche, dans les réseaux à partitions de longue durée, la quantité de paquets mise en tampon dans les nœuds doit être augmentée considérablement pour se remettre dans ces cas. Finalement, les résultats soulignent la prudence dont il faut faire preuve lors de la mise en place de tout mécanisme de fiabilité. Inonder un réseau de NACK et de tentatives de retransmission de paquets aura des effets nuisibles sur le rendement du protocole quand ce réseau est encombré, puisque sa capacité est limitée. Avec le protocole BCAST fiable, chaque nœud surveille le trafic local et supprime les NACK lorsqu'il détecte un trafic trop important et présent depuis un certain temps.

À la dernière étape, les chercheurs ont examiné l'incidence d'une sous-couche de contrôle d'accès au support sur le rendement du meilleur protocole BCAST et du protocole BCAST fiable. Selon le volume du trafic des utilisateurs et la sous-couche de contrôle d'accès au support, la section 7 propose un certain nombre d'indices sur la relation entre la sous-couche de contrôle d'accès au support et la couche de routage. Plus particulièrement, les chercheurs ont remarqué que le protocole BCAST se détériore quand le nombre de collisions de paquets au niveau de la sous-couche de contrôle d'accès au support augmente. Par conséquent, les approches visant à réduire de telles collisions peuvent accroître considérablement le rendement du protocole BCAST. Les résultats démontrent aussi que le modèle de retransmission des paquets fondé sur des NACK augmente le rapport de remise des paquets pour un nombre restreint de scénarios seulement. Le meilleur protocole BCAST se comporte déjà très bien si le trafic des utilisateurs est faible, relativement au débit de données de la sous-couche de contrôle d'accès au support. Par contre, si le réseau devient encombré en raison d'un trafic accru, il faut réduire les NACK afin de prévenir leurs effets négatifs sur le rendement du protocole; il n'y a donc pas beaucoup de différences entre la meilleure version et la version fiable du protocole. Comme le démontre l'analyse de la section 7, la réduction des NACK doit être réalisée minutieusement pour la sous-couche de contrôle d'accès au support, sinon le rendement peut en subir les contrecoups.

En conclusion, le protocole BCAST permet d'obtenir une remise importante des paquets au prix d'une augmentation du temps d'attente des paquets. Le rapport indique que le protocole s'exécute bien pour une vaste gamme de scénarios et un certain nombre de sous-couches de contrôle d'accès au support (qui sont toutes des variantes de la catégorie des protocoles 802.11). La valeur de l'augmentation de la remise des paquets à l'aide d'un modèle de retransmission est toutefois limitée. À mesure que les taux de contrôle d'accès au support augmentent pour les réseaux actuels et futurs, les MANET pourront soutenir un volume considérable de trafic par expéditeur de multidiffusion. Il est possible d'obtenir des rapports élevés de remise de paquets dans ces réseaux en ajustant la quantité de données avec un contrôle de flux afin de demeurer à l'intérieur des paramètres idéaux du protocole et en utilisant le meilleur protocole comme protocole de base.

# Table of contents

# List of figures

## List of tables

# Acknowledgements

This work is funded by the Defence Research and Development Canada (DRDC) and benefited from various discussions with members of the CRC RNS mobile networking group.

# 1. Introduction

Multicasting is the transmission of datagrams (packets) to a group of zero or more hosts identified by a single destination address. A multicast packet is typically delivered to all members of its destination host group with the same reliability as regular unicast packets. In the case of IP Multicasting, for example, the packet is not guaranteed to arrive at all members of the destination group or in the same order relative to other packets.

Multicasting is intended for group-oriented computing. There are more and more applications where one-to-many or many-to-many dissemination is an essential task. The multicast service is critical in applications characterized by the close collaboration of teams (e.g. rescue patrol, battalion, scientists, etc) with requirements for audio and video conferencing and sharing of text and images. In the Internet (IPv4), multicasting facilities were introduced via the Multicast Backbone (MBone), a virtual overlay network on top of the Internet. This overlay network consists of multicast-capable islands connected by tunnels. Each island contains one or more special routers called multicast routers, which are logically connected by these tunnels. These routers manage group membership and cooperate to route data to all hosts wishing to participate in a multicast group. IP multicast groups are identified by special IP addresses. Support for multicasting is an integral component of IPv6, so it can be assumed that multicasting applications will become even more popular with the increased popularity and acceptance of IPv6. Note that the acceptance and use of group-related applications is not only based on technological criteria. [Grudin 2002] for example discusses some sociological issues relevant to the design and use of group applications.

Typically, the membership of a host group is dynamic; that is, hosts may join and leave groups any time. There is no restriction on the location or number of members in a host group. A host may be a member of more than one group at a time. A host does not have to be a member of a group to send packets to it. A host group may be permanent or transient. A permanent group has a well-known, administratively assigned address. It is the address, not the membership of the group that is permanent; at any time a permanent group may have any number of members, even zero. Those IP multicast addresses that are not reserved for permanent groups are available for dynamic assignment to transient groups which exist only as long as they have members. RFC 1700 [RFC 1700] lists well-known multicast addresses for IPv4 as of 1994. More recently, these addresses, like all other well-known (assigned) numbers are managed by an online database, accessible through a web page (currently, www.iana.org). RFC 3171 [RFC 3171] documents the guidelines employed but IANA, the Internet Assigned Numbers Authority, in assigning such well-known IPv4 multicast addresses. Based on those well-known multicast addresses, RFC 2375 [RFC 2375] suggests similar well-known multicast IP addresses for IPv6. RFC 2908 [RFC 2908] proposes a general multicast address allocation architecture for the Internet, and is intended to be generic enough to apply to both IPv4 and IPv6 environments. RFC 3306 [RFC 3306] introduces encoded information in the multicast address to allow for dynamic allocation of IPv6 multicast addresses and IPv6 source-specific multicast addresses. Finally, RFC 3307 [RFC 3307] specifies guidelines for allocating permanent IPv6 multicast addresses, dynamic IPv6 multicast addresses, and permanent IPv6 multicast group identifiers. The purpose of these guidelines is to reduce the probability of IPv6 multicast address collision, not only at the IPv6 layer, but also at the link-layer of media that encode portions of the IP layer address into the link-layer address.

The use of multicasting within a network has many benefits. Multicasting reduces the communication costs for applications that send the same data to multiple recipients [Varshney 2002]. Instead of sending via multiple unicasts, multicasting minimizes the link bandwidth consumption, sender and router processing, and delivery delay. In addition, multicasting provides a simple yet robust communication mechanism whereby a receiver's individual address is unknown or changeable transparently to the source.

Maintaining group membership information and building an optimal multicast distribution structure (typically in the form of a routing tree) is challenging even in wired networks. A very detailed survey of the work done in that area and a discussion of various design trade-offs can be found in [Li 2002]. However, nodes are increasingly mobile. One particularly challenging environment for multicast is a mobile ad-hoc network (MANET). A MANET consists of a dynamic collection of nodes with sometimes rapidly changing multi-hop topologies that are composed of relatively low-bandwidth wireless links. There is no assumption of an underlying fixed infrastructure. Nodes are free to move arbitrarily. Since each node has a limited transmission range, not all messages may reach all the intended hosts. To provide communication through the whole network, a source-to-destination path could pass through several intermediate neighbour nodes. Unlike typical wireline routing protocols, ad-hoc routing protocols must address a diverse range of issues. The network topology can change randomly and rapidly, at unpredictable times. Since wireless links generally have lower capacity, congestion is typically the norm rather than the exception. The majority of nodes will rely on batteries, thus routing protocols must limit the amount of control information that is passed between nodes. Also, multicast group members and other nodes move, thus precluding the use of a fixed multicast topology. [Kunz 2002] provides an overview of some best-effort IP multicast routing protocols for fixed networks and the evolution of these protocols as hosts and finally all nodes (including intermediate routers) become mobile.

The goal of MANETs is to extend mobility into the realm of autonomous, mobile, wireless domains, where a set of nodes form the network routing infrastructure in an ad-hoc fashion. The majority of applications for the MANET technology are in areas where rapid deployment and dynamic reconfiguration are necessary and the wireline network is not available. These include military battlefields, emergency search and rescue sites, classrooms, and conventions where participants share information dynamically using their mobile devices. These applications lend themselves well to multicast operation. In addition, within a wireless medium, it is even more crucial to reduce the transmission overhead and power consumption. Multicasting can improve the efficiency of the wireless link when sending multiple copies of messages by exploiting the inherent broadcast property of wireless transmission.

RFC 3170 [RFC 3170] describes the challenges involved with designing and implementing multicast applications. The document lists a number of multicast applications and derives unique multicast service requirements for various groups of applications. While many applications, such as audio/video distribution, can tolerate loss of data content, many other applications cannot. In addition, even loss-tolerant applications will suffer a performance penalty: an audio stream may experience a short gap or lower fidelity in the presence of loss. Among the loss intolerant application categories are file distribution and caching, monitoring applications (stock prices, sensor readings, etc.), synchronized resources (directories, distributed databases, etc.), concurrent processing, collaboration/shared document editing, and online auctions. A similar discussion of multicast applications and their requirements can be found in [Varshney 2002]. Some of the loss-intolerant applications discussed in these documents are relevant in a MANET environment as well (such as the collaboration, caching/file distribution, or monitoring applications). In addition,

MANET-specific applications such as military command-and-control applications also require a high degree of reliability.

This report is organized as follows. Section 2 reviews alternative definitions of the term “reliable multicast” and justifies the one we have chosen as the basis for our work. Section 3 discussed related work that shows that achieving high packet delivery ratios in MANET multicasting is hard. Section 4 evaluates a number of potentially solutions to the problem of efficiently supporting one-to-many and many-to-many communication. The results show that one particular broadcast protocol, BCAST, appears to be a promising candidate, as it already provides high packet delivery ratios in many scenarios. Section 5 discusses general design alternatives when designing a reliable multicast protocol and explains our choices. Section 6 describes our modifications to BCAST to increase the packet delivery ratio and discusses the resulting protocol performance. Finally, Section 7 provides some insight into the performance of BCAST (both the best-effort version and the reliable version) over different MAC layers, and Section 8 summarizes the key findings of our research.

# 2. Definition of reliable multicasting

As expressed by RFC 2357, “the meaning of reliability varies in the context of different applications” [RFC 2357, page 2]. Consequently, we need to define what exactly we mean by “reliable multicast”. There are a number of possible definitions. In [Sankarasubramaniam 2003], for example, reliable data delivery in a sensor network is defined as receiving enough event notifications from sensors (above a threshold value) to deduce with high probability that an event did indeed occur. Such definitions are very application-specific. More general definitions are discussed in [Li 2002] and elsewhere, where at least three different levels of reliable data packet delivery are distinguished (listed in increasing order of difficulty):

- all data packets are delivered,
- the causal order between data packets is maintained, or
- a total order of data packet delivery is achieved.

All three possible definitions require at least that once a multicast sender has sent a data packet, all multicast receivers will (eventually) receive it. However, there is no guarantee that for multiple data packets sent by the same or different senders, a specific order of reception is maintained. For example, if sender S sends two packets M1 and M2, it is okay for receiver R1 to receive M1 before M2, while receiver R2 may receive those same two packets in the reverse order (M2 before M1).

While delivery of all packets can be seen as a minimum requirement for any reliable multicast mechanism, some applications have more stringent requirements. In particular, the order in which multicast packets are received may matter. An example could be a replicated database, where packets trigger update operations on the data. If two receivers receive and process these update messages in different orders, the replicated data easily becomes inconsistent. For example, assume we have an integer variable A with initial value of 10. If packet M1 advises each receiver to double the value of A, and packet M2 asks to add 5 to A, R1' s final value, after first processing M1 and then M2, will be 25. Similarly, R2' s final value will be 30, since R2 first adds 5 before doubling the resulting sum.

The difference between the second and third definition of reliable multicast is in the constraints imposed by the reception of different packets. The most rigid definition is the third definition, where the multicast protocol determines an ordering among ALL multicast packets (originating from the same or a different sender), and enforces that every receiver receives all packets in exactly that order. The order in which multicast packets are delivered to all receivers is called their “total order”.

A slightly more relaxed constraint is the second definition, which enforces only what is called a “partial order” on the delivery of multicast packets. In a nutshell, this definition says that if M1 could have influenced the content of M2, all receivers will have to receive M1 before M2. If, however, M1 could not have had any impact on M2, than the order in which these two packets can be received is left open. A packet M1 in this scheme can impact a packet M2 (or, as it is also called, “causally precede it”), if either:

- M2 is sent after M1 by the same sender

- M2 is sent by a sender after it received M1

and the transitive closure of these two cases. In essence, M2 "causally follows" M1 if there is a chain of packets from M1 to M2 that are sent in a sequence. Since this relationship does not enforce an ordering on all messages, it is also called a "partial order" relationship.

In this work, we will focus on ensuring that all packets are delivered, anticipating that this will be a difficult enough problem in MANETs. If additional constraints on packet delivery are necessary, they can be implemented based on reliable packet delivery by adding logical timestamps to messages, as defined in [Fidge 1988, Lamport 1978, Mattern 1989].

Actually, even guaranteeing absolute reliability independent of packet ordering is not a realistic goal. In mobile networks, individual receivers can be disconnected from the network for unpredictable amounts of time. As will be discussed later, implementing reliable protocols involves buffering packets to service retransmission requests. Since buffer space is only finite on any concrete computer, we cannot expect to be able to support arbitrarily long disconnections, with the ensuing requirement to buffer copies of all packets until the disconnected receiver(s) reconnect. The only realistic goal therefore is to provide as high a packet delivery ratio as possible with finite resources.

# 3. Why is reliable multicasting hard?

Achieving reliable packet delivery in a MANET is not trivial. A few simulation studies have explored the performance of MANET multicast routing protocols such as the multicast extensions for AODV and ODMRP [Cheng 2001, Ding 2002, Zhu 2002]. These studies commonly simulated an area of usually 1000 x 1000 meters, populated by 50 mobile nodes. Nodes move according to the "random waypoint" mobility model: initially, nodes are placed randomly within the area. Each node picks a destination and moves to that destination based on a speed that is uniformly distributed between 0 and MAX. Once a node reaches the destination, it pauses for PAUSE seconds, after which the process repeats itself. In all these studies, nodes communicated over an IEEE 802.11 wireless link of 2 Mbps, the radio range was 250m. Only a subset of the MANET nodes joined a single multicast group, with some of these nodes sending fixed size data packets to all other nodes at a constant rate (i.e., CBR traffic).

The results show that the packet delivery ratio does drop below 20% (i.e., only 1 out of every 5 packets is, on average, received by a multicast receiver) for a large number of senders [Ding 2002]. In environments with one or a few senders, packet delivery ratios as low as 25% were observed when the mobility rate increased (nodes moved constantly, with MAX speed set to 20 m/s) [Cheng 2001]. Possible improvements, such as pro-actively predicting link breakage and maintaining the multicast distribution data structure before links break (and therefore packets get lost) can increase the packet delivery ratio, but under high mobility scenario, it is still below 90% [Zhu 2002]. In a nutshell, all these protocols exhibit intolerably high packet loss rates under moderate to high mobility rates. Similarly poor results are shown in [Lee 2000] for a number of other multicast routing protocols.

Building and maintaining a multicast distribution structure (typically a tree or mesh) in a MANET with its highly dynamic topology introduces its own complexities and overheads (control messages, data structures at intermediate nodes, etc.). Based on the above studies, this effort does not necessarily result in good performance; it therefore becomes questionable whether it is indeed worth the effort. Following this line of thought, some researchers have explored whether broadcasting/flooding a MANET with packets could be a viable alternative to ensure high packet delivery ratios. The results in [Obraczka 2001a, Obraczka 2001b] indeed show that flooding results in higher packet delivery ratios then ODMRP, which in turn outperforms AODV. But in the scenarios studied in these papers, flooding could result in packet delivery ratios as low as 70%, leading the authors to conclude that "*even flooding is insufficient for reliable multicast in ad hoc networks when mobility is very high*" [Obraczka 2001b, page 627]. In addition, the simulation scenarios were all based on the assumption that every node in the MANET was interested in the data packets (i.e., a global broadcast). More generally, only a subset of nodes will be interested in any specific multicast group, flooding the data to all nodes may induce a high overhead, negating one of the stated advantages of multicasting (see above).

This network overhead was exacerbated in the two studies by the fact that the packet broadcast was implemented in a trivial manner; with every node re-broadcasting a packet the first time it receives it. As discussed in [Williams 2002] and [Lou 2002], more efficient broadcast algorithms can be implemented. However, even those algorithms will suffer from low packet delivery ratios (60%-80%) as the severity of the network environment (mobility rate, traffic load, etc.) increases.

In conclusion, it seems that flooding/broadcasting data in a MANET is not sufficient to ensure high packet delivery ratios. While flooding is attractive due to its absence of a multicast distribution structure (and its ensuing maintenance), it may lead to high network traffic when propagating data packets to nodes that are not interested in it. However, to explore this issue further, we implemented some best-effort multicast approaches and compared them by running extensive simulations, as discussed next.

# 4. Comparison of multicast approaches

The first in-depth study explored whether multicasting in a MANET context is worthwhile. Due to the dynamic nature of the network, building and maintaining an efficient multicast distribution structure may be inefficient. To support many-to-many communication, we could alternatively use:

- dedicated unicast communication or
- broadcasting.

In the first instance, if a sender communicates with N receivers, it would open N point-to-point flows to the receivers. Note that this requires the sender to know the identities of all receivers. Furthermore, these N routes now have to be maintained as well, so we do not really expect the unicast solution to perform better. However, we include it for completeness sake and to establish a baseline performance. In the latter instance (i.e., using broadcast protocols), the identity and number of receivers can remain unknown to the sender(s). Rather, a data packet is delivered to all nodes in the network, with those nodes interested in the data simply passing it up the protocol stack to the application. A broadcast solution may require little or no routing overhead. However, broadcasting may be inefficient: every node in the network receives every data packet. For sparse multicast groups (only a small subset of nodes in the network are receivers), more efficient ways of transmitting data packets may exist.

To explore whether multicasting can really achieve the claimed advantages, we conducted a thorough simulation study, comparing and evaluating 7 different routing protocols to support the communication between N senders and M receivers. In particular, we studied:

- 2 unicast routing protocols: DSR and AODV
- 3 multicast routing protocols: ADMR, ODMRP, and the multicast extensions to AODV
- 2 broadcast protocols: FLOOD and BCAST

## 4.1. Related work

Many papers present performance results, based on simulations, to study and evaluate protocol behavior under a range of scenarios. Almost any paper presenting a new routing protocol will contain an evaluation section that compares the proposed protocol against a (typically small) set of related protocols. Good examples are [Das 2001], which is co-written by the designers of AODV and compares this protocol with DSR, and [Jetcheva 2001], which introduces ADMR and compares it with ODMRP.

A number of papers are dedicated to performance comparisons of various routing protocols, typically based on simulation. [Broch 1998] is one of the earliest comparative studies of MANET unicast routing protocols in NS2, indicating the superior performance of AODV and DSR. [Bagrodia 2000, Lee 2000] simulate several multicast routing protocols developed specifically for MANET and evaluate them under diverse network scenarios using the GloMoSim library. The reported results show that mesh protocols performed significantly better than the tree protocols in

mobile scenarios. Finally, a number of papers have studied the performance of broadcast protocols, one of the more recent papers is [Williams 2002], which categorizes various broadcast protocols into a small set of categories, implements a representative protocol from each category in a simulator and conducts an in-depth analysis of the performance across a range of scenarios. The results show that all protocols will eventually suffer from low packet delivery ratio as the mobility rate increases, and that BCAST is one of the protocol that will "break" the latest.

All these papers compare only similar protocols with each other: [Broch 1998] focuses on unicast protocols, [Bagrodia 2000, Lee 2000] on multicast protocols, and [Williams 2002] on broadcast protocols. The only efforts we are aware of that compares multicast protocols with broadcast protocols are reported in [Obraczka 2001a, Obraczka 2001b], which show that flooding results in higher packet delivery ratios then ODMRP, which in turn outperforms AODV. However, the simulation scenarios were all based on the assumption that every node in the MANET was interested in the data packets (i.e., a broadcast scenario). And [Lee 2002] provides simulation results that compare flooding with ODMRP and three other multicast routing protocols, but the discussion in the paper focuses mostly on the relative performance of the multicast protocols. This report is the first systematic effort to compare various alternatives to support one-to-many or many-to-many communication in a MANET with a variety of protocols.

## 4.2. Protocol descriptions

All protocols discussed in this section have been developed for MANETs. The first two protocols are on-demand unicast routing protocols, currently considered for standardization by the IETF. The multicast protocols have been proposed by various research groups in recent years for MANETs and follow a design similar to the unicast routing protocol: a packet distribution structure is created and maintained on-demand, the differences are primarily in the nature of the multicast distribution structure. Finally, the two broadcast protocols range from a very trivial one, FLOOD, to a rather complex one, BCAST. The latter minimizes the number of nodes re-broadcasting a data packet while still ensuring that all nodes receive a data packet with high probability.

### 4.2.1 Unicast protocols

Unicast routing in a MANET has attracted a lot of attention and consequently a large number of unicast routing protocols have been proposed. These protocols can broadly be classified into pro-active routing protocols, on-demand routing protocols, and hybrid protocols. In pro-active routing protocols, similar to the routing in the Internet, routes to all possible destinations are maintained at all times, typically by having nodes periodically exchange routing protocol control messages. Example protocols in this category are OLSR (Optimized Link State Routing) or DSDV (Destination-Sequenced Distance Vector protocol). On-demand protocols, on the other hand, only worry about routes to destinations that are actually recipients of data. These routes are discovered "on-demand" using a request-reply cycle. DSR and AODV, discussed below, fall into this category. Finally, hybrid protocols such as ZRP (Zone Routing Protocol) combine aspects of the first two categories, pro-actively maintaining routes to "close" nodes and discovering routes to "remote" nodes on-demand.

Existing simulation studies indicate that in scenarios with high mobility and relatively few senders/receivers, on-demand protocols outperform pro-active protocols. The

scenarios we are investigating match this description: we have relatively few senders, and the network is highly dynamic. We therefore selected two on-demand routing protocols for this study.

DSR [Johnson 2001] is based on the concept of source routing, a routing technique in which the sender of the packet determines the complete sequence of the nodes through which to forward the packet. The sender explicitly lists this route in the packet's header, identifying each forwarding "hop" by the address of the next node to which to transmit the packet on its way to the destination host. The DSR protocol consists of two mechanisms: Route Discovery and Route Maintenance. When a mobile node wants to send a packet to some destination, it first checks its route cache to determine whether it already has a route to the destination. If it has one, it will use this route to send the packet. Otherwise, it will initiate route discovery by broadcasting a route request packet. When receiving a request packet, a node appends its own address to the route record in the route request packet if it did not receive this request message before, and re-broadcasts the query to its neighbors. Alternatively, it will send a reply packet to the source without propagating the query packet further if it can complete the query from its route cache. Furthermore, any node participating in route discovery can learn routes from passing packets and gather this routing information into its route cache. When sending or forwarding a packet to a destination, Route Maintenance is used to detect if the network topology has changed such that the link used by this packet is broken. Each node along the route, when transmitting the packet to the next hop, is responsible for detecting if its link to the next hop has broken. When the retransmission and acknowledgement mechanism detects that the link is broken, the detecting node returns a Route Error packet to the source of the packet. The node will then search its route cache to find if there is an alternative route to the destination of this packet. If there is one, the node will change the source route in the packet header and send it using this new route. This mechanism is called "salvaging" a packet. When a Route Error packet is received or overheard, the link in error is removed from the local route cache and all routes, which contain this hop, must be truncated at that point. The source can then attempt to use any other route to the destination that is already in its route cache, or can invoke Route Discovery again to find a new route.

In AODV [Perkins 1999], every node maintains a route table. Every entry of the table has source and destination sequence numbers and other soft-state information. When a source node needs to send a packet to a destination node for which it has no routing information in its table, the Path Discovery process is initiated. The source node broadcasts a route request (RREQ) to its neighbors. Neighbors either can reply with a route reply (RREP) if they have a route to the destination or rebroadcast the RREQ. Each node remembers only the next hop and not the entire route, as in source routing. Once the next hop becomes unreachable, the node upstream of the break propagates an unsolicited RREP with a fresh sequence number and infinity hop count to all active upstream neighbors. Those nodes subsequently relay that message to their active neighbors. This process continues until all active source nodes are notified. Upon receiving notification of a broken link, source nodes can restart the discovery process if they still require the destination.

### 4.2.2 Multicast protocols

All multicast routing protocols create paths to other hosts on demand. The idea is based on a query-response mechanism similar to reactive unicast routing protocols. In the query

phase, a node explores the environment. Once the query reaches the destination, the response phase is entered and establishes the path. The following three multicast protocols are all based on this approach. The difference is in the type of multicast distribution structure (mesh versus tree) and whether there is one shared structure for the multicast group or one per source node.

The multicast extensions for the AODV (Ad-hoc On-Demand Distance Vector**)** routing protocol [Royer 1999] discover multicast routes on demand using a broadcast route-discovery mechanism. The protocol builds a shared multicast tree based on hard state, repairing broken links and explicitly dealing with network partitions. A mobile node originates a Route Request (RREQ) message when it wishes to join a multicast group, or when it has data to send to a multicast group but it does not have a route to that group. If an intermediate node receives a RREQ and it does not have a route to that group, it rebroadcasts the RREQ to its neighbors. As the RREQ is broadcast across the network, nodes set up pointers to establish the reverse route in their route tables. If a node receives a RREQ for a multicast group, it may reply if its recorded sequence number for the multicast group is at least as great as that contained in the RREQ. The responding node updates its route and multicast route tables by placing the requesting node's next hop information in the tables, and then unicasts a Request Response (RREP) back to the source node. As nodes along the path to the source node receive the RREP, they add both a route table and a multicast route table entry for the node from which they received the RREP, thereby creating the forward path. When a source node broadcasts a RREQ for a multicast group, it often receives more than one reply. The source node keeps the received route with the greatest sequence number and shortest hop count to the nearest member of the multicast tree for a specified period of time, and disregards other routes. At the end of this period, it enables the selected next hop in its multicast route table, and unicasts an activation message (MACT) to this selected next hop. The next hop, on receiving this message, enables the entry for the source node in its multicast route table. This process continues until the node that originated the RREP (member of tree) is reached. The activation message ensures that the multicast tree does not have multiple paths to any tree node. Nodes only forward data packets along activated routes in their multicast route tables.

The first member of the multicast group becomes the leader for that group. A node assumes the group leadership role after unsuccessfully attempting to join a multicast group multiple times. The multicast group leader is responsible for maintaining the multicast group sequence number and broadcasting this number to the multicast group. This is done through a Group Hello message. The Group Hello contains extensions that indicate the multicast group IP address and sequence numbers (incremented every Group Hello) of all multicast groups for which the node is the group leader. Nodes use the Group Hello information to update their request table.

Since MAODV keeps hard state in its routing table, the protocol has to actively track and react to changes in this tree. If a member terminates its membership with the group, the multicast tree requires pruning. Links in the tree are monitored to detect link breakages. When a link breakage is detected, the node that is further from the multicast group leader (downstream of the break) is responsible for repairing the broken link. If the tree cannot be reconnected, a new leader for the disconnected downstream node is chosen as follows. If the node that initiated the route rebuilding is a multicast group member, it becomes the new multicast group leader. On the other had, if it was not a group member and has only

one next hop for the tree, it prunes itself from the tree by sending its next hop a prune message. This continues until a group member is reached. Once these two partitions reconnect, a node eventually receives a Group Hello for the multicast group that contains group leader information that differs from the information it already has. If this node is a member of the multicast group, and if it is a member of the partition whose group leader has the lower IP address, it can initiate reconnection of the multicast tree.

The second multicast protocol that builds and maintains a multicast distribution structure is ODMRP: On-Demand Multicast Routing Protocol [Lee 1999, Lee 2002]. ODMRP is mesh-based, and uses a forwarding group concept (only a subset of nodes forwards the multicast packets). In ODMRP, group membership and multicast routes are established and updated by the source on demand. When a multicast source has packets to send, but no route to the multicast group, it broadcasts a Join-Query control packet to the entire network. This Join-Query packet is periodically broadcast to refresh the membership information and update routes. When an intermediate node receives the Join-Query packet, it stores the source ID and the sequence number in its message cache to detect any potential duplicates. The routing table is updated with the appropriate node ID (i.e. backward learning) from which the message was received for the reverse path back to the source node. If the message is not a duplicate and the Time-To-Live (TTL) is greater than zero, it is rebroadcast. When the Join-Query packet reaches a multicast receiver, it creates and broadcasts a "Join Reply" to its neighbors. When a node receives a Join Reply, it checks if the next hop node ID of one of the entries matches its own ID. If it does, the node realizes that it is on the path to the source and thus is part of the forwarding group and sets the FG_FLAG (Forwarding Group Flag). It then broadcasts its own Join Table built upon matched entries. The next hop node ID field is filled by extracting information from its routing table. In this way, each forward group member propagates the Join Reply until it reaches the multicast source via the selected path (shortest). This whole process constructs (or updates) the routes from sources to receivers and builds a mesh of nodes, the forwarding group. These meshes are source-based: in an environment with many senders, a number of these meshes will have to be built and maintained. On the other hand, no work is required to update the mesh as the topology changes or nodes join/leave the multicast group: such changes get reflected the next time the mesh is rebuilt.

The Adaptive Demand-Driven Multicast Routing (ADMR) protocol operates entirely in an on-demand fashion [Jetcheva 2001]. The protocol eliminates any periodic elements that exist in protocols such as ODMRP (the Join-Query) or MAODV (the Group Hello message). A source-based forwarding tree is created whenever there is at least one source and one receiver in the network (i.e., there is a multicast distribution tree for each active source). AMDR monitors the traffic pattern of the sources to detect link breakages and inactive sources. In the former case, a local link repair is initiated, in the latter case the multicast distribution structure silently expires. To distinguish the two cases, keep-alive messages are transmitted at increasing inter-packet times. In addition, ADMR resorts to flooding data packets in one of the following two cases:

- Occasionally, a data packet is flooded to recover from network partitions (this is an optional protocol feature).

- Flooding is the preferred data delivery mechanism if ADMR detects a high mobility rate. In this case, the protocol assumes that the multicast state cannot be setup and maintained in a timely fashion with low overhead. Once in flooding mode, the

protocol periodically reverts to multicast routing to explore whether the mobility rate has decreased.

Flooding is implemented based on the idea of a forwarding group, similar to ODMRP.

### 4.2.3 Broadcast protocols

Broadcasting protocols deliver data to all nodes in a network, independent of whether they are interested in that data or not. Since even unicast and multicast routing protocols often have a broadcast component (for example, the route discovery phase in on-demand unicast routing protocols), efficient broadcast protocols have been investigated heavily. [Williams 2002] gives an overview of the various categories of broadcast protocols and provides simulation results under various mobility scenarios. For the purpose of this study, we selected two broadcast protocols: a very simple protocol (FLOOD) and one of the more complex protocols (BCAST). Based on the results presented in [Williams 2002], we expect BCAST to outperform FLOOD.

The first, and simplest protocol is FLOOD. It essentially implements standard flooding: each node, upon receiving a packet for the first time, will re-broadcast it over its wireless interface (i.e., using MAC-layer broadcasting). To reduce the chance of packet collisions, re-broadcasts are randomly jittered by 10 ms.

The second protocol, BCAST, implements a scalable broadcast algorithm similar to the algorithm described in [Lou 2002]. BCAST uses 2-hop neighbor knowledge that is exchanged by periodic "Hello" messages. Each "Hello" message contains the node's identifier (IP address) and its list of known neighbors. After a node receives a "Hello" packet from all its neighbors, it has two-hop topology information. If node B receives a broadcast from node A, B knows all neighbors of A. If B has neighbors not covered by A, it schedules the broadcast packet with a random delay. If, during the delay, B receives another copy of this broadcast from C, it can check whether its own broadcast will still reach new neighbors. If this is no longer the case, it will drop the packet. Otherwise, the process continues until B's timer goes off and B itself rebroadcasts the packet.

One issue to be solved is the determination of the random delay. The original authors of that protocol suggested a dynamic strategy. Each node searches its neighbor table for the maximum neighbor degree of any neighbor node, MAX. If its own node degree is N, it calculates the random delay as MAX/N. This is a greedy strategy: nodes with the most neighbors usually broadcast before others.

## 4.3. Simulation environment

To compare the performance of the various "multicast" solutions, we studied the above protocols in NS2. Except where noted, all results are based on version 2.1b9a, running under Red Hat Linux release 7.3. To this end, we set up a rather challenging simulation environment, using the following parameters (similar to other setups reported in the literature):

- Area: 1500 x 300 meters
- Number of nodes: 50

- Simulation length: 910 seconds
- Number of repetitions: 10
- Physical/Mac layer: IEEE 802.11 at 2 Mbps, 250 meter transmission range

Mobility model: random waypoint model with no pause time, maximum speed either 20 m/s (high mobility scenarios) or 1 m/s (low mobility scenarios). Some additional experiments with maximum speeds of 15 m/s, 10 m/s, 5 m/s were also done. The results are not reported here.

All nodes are in constant movement in our experiments. The only traffic is the multicast traffic. We study a range of multicast senders (1, 2, 5, or 10), sending to a number of multicast receivers (10, 20, 30, 40, or 50). We keep the sender and receiver sets disjoint. For example, in a scenario with 10 senders and 30 receivers, nodes 0 through 9 are the senders and nodes 20 through 49 are the receivers. Only in scenarios with 50 multicast receivers will some nodes act as both sender and receiver. In these scenarios, we expect the packet delivery ratio to be slightly better, since packet delivery within a single node is not subject to network problems.

All receivers join the single multicast group at the beginning of the simulation; the sender(s) start sending data 30 seconds into the simulation (so where appropriate, the routing protocol can start building its multicast distribution structure, for example in MAODV). After 900 seconds, all senders stop transmitting data, during the remaining 10 seconds, data packets still in flight have a chance to be delivered. We decided to allow for this additional 10 seconds in our simulations to avoid the problem of how to account correctly for packets that are in flight at simulation end. In a multicast tree, for example, the receivers in a subtree could not have received a packet that is currently being handled at an interior node (the subtree root). To deduct this number from the number of expected packets to be received, we would need to know the exact topology of the routing structure.

Each sender sends data at a specified rate and size. To explore different traffic loads, the following three different traffic sources per sender were evaluated:

- 2 packets per second, each packet 256 bytes long (light traffic)
- 4 packets per second, each packet 512 bytes long (medium traffic)
- 8 packets per second, each packet 1024 bytes long (heavy traffic)

However, under the latter two loads, the MANET is almost always heavily congested, resulting in very poor protocol performance. We therefore present the results for the light traffic load only. Some work on increasing the overall network capacity to support higher offered loads is currently under way.

A total of seven routing protocols are studied. Implementations of AODV and DSR are provided with the NS2 distribution, and we used the NS2 version 2.1b9a implementations without any modification. In the traffic files, each sender initiates N unicast connections to the multicast receivers, generating the same amount of traffic (2 packets per second, each packet 256 bytes in size) for each connection.

Cheng [Cheng 2001] and Zhu [Zhu 2002] implemented MAODV. Zhu also added pro-active tree maintenance to improve packet delivery further; this feature was enabled in the results reported

here. The MONARCH research group provides ODMRP and ADMR implementations for NS2 version 2.1b8 [Monarch 2003], and again we used them without modification. This NS2 version supports the same scenario files as version 2.1b9a, so the scenarios are the same across simulator versions. The results are based on the most recent protocol implementations, including the March 2003 modifications.

We implemented the two broadcast protocols, FLOOD and BCAST, in NS2 version 2.1b9a. FLOOD is a trivial protocol: each node, upon receiving a broadcast, checks whether it received that packet before. If not, it will re-broadcast the packet, jittering the transmission by 10 ms to reduce the likelihood of collision with neighboring nodes. For nodes that also run a multicast receiver, a copy of that packet is passed up to the receiving agent.

BCAST is a more complicated protocol. Nodes periodically broadcast HELLO messages to exchange neighborhood information. In our implementation, HELLO messages are scheduled with uniform distribution in the interval [1.5 seconds, 2.5 seconds]. A node is not considered a neighbor anymore if we miss the next HELLO message. Preliminary runs indicate that these parameter settings result in good performance. Packet broadcast is delayed to allow for a more aggressive drop strategy. As discussed above, the idea is to allow nodes with many neighbors to broadcast first by using a scaling factor of MAX/N. To explore the effect of this factor, we scaled this factor with 100 ms and 10 ms. In the former case, packet latencies will increase, but there is better chance that packet broadcasts are cancelled, resulting in lower overheads and potentially higher packet delivery ratios.

The basic performance metrics are Packet Delivery Ratio (PDR) and Packet Latency. Packet delivery ratio is defined as the percentage of received packets, relative to the total number of packets ideally received. Counting the total number of received packets is straightforward in the simulations. The total number of packets ideally received is the number of packets sent by the sender(s) in the case of unicast routing protocols. In the case of multicast or broadcast protocols, the number of intended multicast receivers multiplies the number of packets sent. Packet latency is the elapsed time between a packet being transmitted and ultimately received. We only measure packet latency for those packets that are received at a multicast receiver. Also, both metrics are calculated as averages over all senders and receivers, we do not break them down by a specific sender or receiver.

An ideal protocol will achieve high packet delivery ratio and low packet latency. It will also do this with little overhead. Traditionally, counting the number of control messages and relating them to the number of received packets measures protocol overhead. However, FLOOD does not generate any dedicated control messages. And the overheads in any broadcast protocol are not only related to any control messages, but also to the waste of delivering packets to nodes not interested in this data. So we generalize the protocol overhead, defining metrics that capture the "network efficiency of the protocol":

- Packet send ratio (PSR): the number of packet transmissions (at the MAC layer) per data packet received by a multicast receiver

- Bytes send ratio (BSR): the number of bytes transmitted (at the MAC layer) per data packet received by a multicast receiver.

These metrics capture the normalized total traffic in the network. The PSR, ideally, could be very small. Since for many protocols we broadcast packets at the MAC layer, multiple receivers pick up that packet (depending on the geographical distribution of senders and receivers), so that

metric could be as low as 1/n, where n is the number of multicast receivers. However, control messages and suboptimal re-broadcasting of packets will increase this metric. In flooding, for example, each node will re-broadcast a data packet. If the number of multicast receivers is smaller than the number of nodes in the network, this metric will be greater than 1.

We will also refer to the BSR where appropriate, since not all packets are of the same size. The various unicast and multicast routing protocols will transmit control messages of varying size, though typically smaller than the size of a data packet. In addition, for the unicast protocols, which transmit data (and some control) packets in unicast mode, the PSR at the MAC layer is inflated by a factor of 4: each unicast packet will be transmitted after an RTS/CTS exchange, followed by an ACK. In our analysis, we do not separate these out and simply count all packet transmissions at the MAC layer when calculating the PSR.

## 4.4. Simulation results

The following sections outline the results for the various unicast, multicast, and broadcast protocols. For the most part, we focus on packet delivery ratio (PDR) and packet latency. The additional metrics will come into play only when comparing protocols of similar packet delivery performance. All results are presented in table form, with the number of senders across the columns and the number of receivers across the rows.

### 4.4.1 Unicast protocols

***Table 1: PDR and latency for DSR, 1 m/s maximum speed***

| | 1 Sender | | 2 Senders | | 5 Senders | | 10 Senders | |
|---|---|---|---|---|---|---|---|---|
| | PDR | Latency | PDR | Latency | PDR | Latency | PDR | Latency |
| **10 Receivers** | 0.973 | 0.040 | 0.970 | 0.025 | 0.754 | 1.054 | 0.331 | 2.536 |
| **20 Receivers** | 0.983 | 0.202 | 0.888 | 0.520 | 0.303 | 2.601 | 0.149 | 3.411 |
| **30 Receivers** | 0.642 | 1.653 | 0.436 | 1.502 | 0.168 | 2.904 | 0.082 | 3.441 |
| **40 Receivers** | 0.363 | 2.376 | 0.213 | 2.231 | 0.083 | 3.555 | 0.039 | 3.202 |
| **50 Receivers** | 0.254 | 2.410 | 0.152 | 2.142 | 0.073 | 2.561 | 0.048 | 1.655 |

***Table 2: PDR and latency for DSR, 20 m/s maximum speed***

| | 1 Sender | | 2 Senders | | 5 Senders | | 10 Senders | |
|---|---|---|---|---|---|---|---|---|
| | PDR | Latency | PDR | Latency | PDR | Latency | PDR | Latency |
| **10 Receivers** | 0.712 | 0.805 | 0.326 | 1.946 | 0.140 | 2.753 | 0.085 | 2.911 |
| **20 Receivers** | 0.205 | 2.964 | 0.124 | 3.199 | 0.069 | 2.835 | 0.040 | 2.779 |
| **30 Receivers** | 0.146 | 3.011 | 0.074 | 3.166 | 0.037 | 3.271 | 0.023 | 2.903 |
| **40 Receivers** | 0.097 | 3.306 | 0.049 | 3.458 | 0.025 | 3.243 | 0.015 | 2.844 |
| **50 Receivers** | 0.093 | 2.495 | 0.063 | 2.276 | 0.039 | 1.630 | 0.032 | 1.012 |

Using DSR to deliver data to multiple recipients is not very attractive, even at low mobility. The packet delivery ratio is high for the 1 or 2 sender cases and relatively few multicast receivers only, with reasonable latencies. Using DSR for many-to-many communication, however, does not scale at all: as the network becomes more dynamic (maximum speed 20 m/s), or the number of multicast senders or receivers increases

(resulting in an increase in unicast connections), the packet delivery ratio drops drastically, with packet latency increasing to seconds. For scenarios with relatively many senders (5 or 10 senders), overall protocol performance is extremely poor. For scenarios with a comparable number of unicast connections, such as 1 sender and 40 receivers vs. 2 senders and 20 receivers, or 5 senders and 20 receivers vs. 10 senders and 10 receivers, the scenarios with the higher number of senders typically perform better. This is reasonable: with more senders, the traffic sources are more spread throughout the MANET. Since a single node generates fewer data packets in these cases, they are less likely to cause packet loss due to network congestion right at the source.

The increased performance in the 10 senders/50 receivers cases (and to a lesser extent in the 5 senders/50 receivers cases) is due to the fact that 2% of all packets are delivered within a node (since the same host runs a multicast sender and a multicast receiver). With relatively few packets delivered otherwise, these packets increases PDR and significantly reduce packet latency for delivered packets.

In most cases, between 12-88 packets are transmitted at the MAC layer for each data packet delivered under low mobility. This number rises to 68 to 218 for the high mobility scenarios. Even accounting for the fact that 70% or so of these packets are MAC layer control packets, DSR seems to be a rather expensive way to multicast packets.

***Table 3: PDR and latency for AODV, 1 m/s maximum speed***

| | 1 Sender | | 2 Senders | | 5 Senders | | 10 Senders | |
|---|---|---|---|---|---|---|---|---|
| | PDR | Latency | PDR | Latency | PDR | Latency | PDR | Latency |
| **10 Receivers** | 0.995 | 0.022 | 0.996 | 0.024 | 0.756 | 1.430 | 0.277 | 4.543 |
| **20 Receivers** | 0.996 | 0.033 | 0.966 | 0.208 | 0.328 | 2.957 | 0.128 | 3.676 |
| **30 Receivers** | 0.988 | 0.091 | 0.730 | 0.942 | 0.209 | 2.839 | 0.078 | 3.081 |
| **40 Receivers** | 0.915 | 0.321 | 0.537 | 1.159 | 0.147 | 2.725 | 0.055 | 2.650 |
| **50 Receivers** | 0.855 | 0.420 | 0.452 | 1.118 | 0.135 | 2.217 | 0.061 | 1.734 |

***Table 4: PDR and latency for AODV, 20 m/s maximum speed***

| | 1 Sender | | 2 Senders | | 5 Senders | | 10 Senders | |
|---|---|---|---|---|---|---|---|---|
| | PDR | Latency | PDR | Latency | PDR | Latency | PDR | Latency |
| **10 Receivers** | 0.974 | 0.038 | 0.972 | 0.040 | 0.499 | 1.430 | 0.199 | 2.092 |
| **20 Receivers** | 0.974 | 0.049 | 0.784 | 0.558 | 0.219 | 1.920 | 0.095 | 1.915 |
| **30 Receivers** | 0.939 | 0.167 | 0.492 | 1.213 | 0.135 | 1.932 | 0.060 | 1.781 |
| **40 Receivers** | 0.826 | 0.434 | 0.342 | 1.361 | 0.092 | 1.926 | 0.041 | 1.688 |
| **50 Receivers** | 0.688 | 0.734 | 0.279 | 1.298 | 0.089 | 1.516 | 0.050 | 1.071 |

The performance of AODV by and large is similar to the performance of DSR: as the number of sender and/or multicast receiver increases (resulting in an increase in the number of unicast connections), the PDR drops and packet latency increases. Again, for 10 senders scenarios, with 50 receivers (i.e., all nodes receive the multicast data), PDR and latency are slightly improved, since now about 2% of all packets are delivered locally (i.e., within a node). The performance under high-speed scenarios is generally worse than under low-speed scenarios.

AODV does perform better than DSR and behaves more consistently. The network efficiency, in terms of MAC packets transmitted per data packet delivered (PSR), ranges from 11 to 52 packets under low mobility scenarios and from 15 to 68 packets under high mobility scenarios. Exploring the trace files in more depth, roughly about 70% of all packet transmissions at the MAC layer are MAC layer control packets (RTS, CTS, and ACK). If we ignore these control packets, the revised PSR (counting only network layer packets) ranges from 3.3 to 15.6 for low mobility scenarios and 4.5 to 20.8 for high mobility scenarios.

Unicast protocols are, in general, not really a good choice when multicasting data. In particular when dealing with more than 1 or 2 senders, the overall protocol performance is rather poor, due to the high traffic load injected into the network. Mobility also has a negative impact on overall performance: high mobility scenarios uniformly have lower PDR and higher packet latency than low mobility scenarios. For 1-to-many multicasting, AODV seems to perform reasonably well, achieving packet delivery ratios of close to 99% or better under low mobility and around 97% under high mobility with packet latencies of a few tens of milliseconds. This performance, however, drops as the number of multicast receivers increases to a sizeable fraction of the number of nodes (30 or more nodes). For many-to-many communication, PDR is extremely low even for a relatively small number of senders. Overall, unicast routing protocols are therefore a poor choice to support a wide range of many-to-many communication scenarios.

### 4.4.2 Multicast protocols

The results presented in the preceding section serves as a base case: in the absence of any multicast or broadcast protocol, the only way to deliver data to multiple recipients is to set up dedicated unicast connections. This section summarizes the results from applying three distinct multicast routing protocols: MAODV, ODMPR, and AMDR.

***Table 5: PDR and latency for MAODV, 1 m/s maximum speed***

| | 1 Sender | | 2 Senders | | 5 Senders | | 10 Senders | |
|---|---|---|---|---|---|---|---|---|
| | PDR | Latency | PDR | Latency | PDR | Latency | PDR | Latency |
| **10 Receivers** | 0.985 | 0.033 | 0.990 | 0.041 | 0.940 | 0.067 | 0.544 | 0.510 |
| **20 Receivers** | 0.990 | 0.048 | 0.975 | 0.059 | 0.843 | 0.154 | 0.433 | 0.837 |
| **30 Receivers** | 0.972 | 0.064 | 0.977 | 0.079 | 0.721 | 0.507 | 0.363 | 1.238 |
| **40 Receivers** | 0.981 | 0.080 | 0.952 | 0.101 | 0.608 | 0.800 | 0.309 | 1.611 |
| **50 Receivers** | 0.981 | 0.089 | 0.943 | 0.115 | 0.583 | 0.890 | 0.321 | 1.583 |

***Table 6: PDR and latency for MAODV, 20 m/s maximum speed***

| | 1 Sender | | 2 Senders | | 5 Senders | | 10 Senders | |
|---|---|---|---|---|---|---|---|---|
| | PDR | Latency | PDR | Latency | PDR | Latency | PDR | Latency |
| **10 Receivers** | 0.888 | 0.039 | 0.916 | 0.049 | 0.845 | 0.110 | 0.461 | 0.643 |
| **20 Receivers** | 0.903 | 0.056 | 0.925 | 0.072 | 0.708 | 0.275 | 0.358 | 0.982 |
| **30 Receivers** | 0.882 | 0.074 | 0.908 | 0.096 | 0.598 | 0.514 | 0.301 | 1.215 |
| **40 Receivers** | 0.898 | 0.091 | 0.881 | 0.124 | 0.509 | 0.833 | 0.261 | 1.382 |
| **50 Receivers** | 0.902 | 0.101 | 0.863 | 0.148 | 0.491 | 0.852 | 0.275 | 1.295 |

MAODV performs significantly better than the unicast routing protocols. The MAODV performance is particularly high under low mobility for few (1 or 2 senders), resulting in PDRs of around 98% and latencies of a few tens of milliseconds. Exploring the trace files in more detail, we noticed that the single shared tree, with all the operations to maintain its hard state, becomes the bottleneck in our scenarios at high mobility and/or for a large number of multicast senders.

Looking at the network efficiency, scenarios with relatively few multicast receivers generate more relative overhead than scenarios with many multicast receivers, where the cost of building and maintaining the multicast tree is amortized over an increased number of packet deliveries. However, with only one, shared, multicast tree being built and maintained, the overhead grows relatively slow with an increase in the number of multicast senders (having more multicast senders will incur a higher overhead since they all need to find and maintain a path to the multicast tree). In low mobility scenarios, PSR ranges from 7.77 for 1 sender/10 receivers to 11.0 for 10 senders/10 receivers. This number drops to 4.6 for all scenarios with 50 multicast receivers (all nodes are part of the multicast tree). In high mobility scenarios, the overhead is higher, ranging from 11.65 in the 1sender/10 receivers scenarios to 13.75 for the 10 senders/10 receivers cases. For the 50 receivers scenarios under high mobility, this ratio drops to around 6. However, since almost all MAC transmissions are unicast, again a high fraction (approximately 70%) of these packets are MAC layer control packets (RTS, CTS, and ACK).

***Table 7: PDR and latency for ODMRP, 1 m/s maximum speed***

| | 1 Sender | | 2 Senders | | 5 Senders | | 10 Senders | |
|---|---|---|---|---|---|---|---|---|
| | PDR | Latency | PDR | Latency | PDR | Latency | PDR | Latency |
| **10 Receivers** | 0.995 | 0.009 | 0.991 | 0.010 | 0.973 | 0.015 | 0.937 | 0.031 |
| **20 Receivers** | 0.995 | 0.010 | 0.991 | 0.011 | 0.974 | 0.017 | 0.936 | 0.039 |
| **30 Receivers** | 0.993 | 0.011 | 0.990 | 0.012 | 0.972 | 0.017 | 0.934 | 0.043 |
| **40 Receivers** | 0.992 | 0.011 | 0.990 | 0.012 | 0.972 | 0.018 | 0.932 | 0.049 |
| **50 Receivers** | 0.973 | 0.011 | 0.971 | 0.012 | 0.953 | 0.018 | 0.915 | 0.056 |

***Table 8: PDR and latency for ODMRP, 20 m/s maximum speed***

| | 1 Sender | | 2 Senders | | 5 Senders | | 10 Senders | |
|---|---|---|---|---|---|---|---|---|
| | PDR | Latency | PDR | Latency | PDR | Latency | PDR | Latency |
| **10 Receivers** | 0.987 | 0.010 | 0.992 | 0.011 | 0.977 | 0.016 | 0.940 | 0.033 |
| **20 Receivers** | 0.991 | 0.010 | 0.992 | 0.011 | 0.979 | 0.017 | 0.937 | 0.039 |
| **30 Receivers** | 0.991 | 0.010 | 0.994 | 0.012 | 0.980 | 0.017 | 0.935 | 0.046 |
| **40 Receivers** | 0.993 | 0.010 | 0.994 | 0.012 | 0.979 | 0.018 | 0.934 | 0.052 |
| **50 Receivers** | 0.973 | 0.010 | 0.974 | 0.012 | 0.961 | 0.018 | 0.916 | 0.060 |

ODMRP shows better protocol performance than the unicast protocols: the packet delivery ratio is consistently higher and latency is significantly reduced (often only a few milliseconds). ODMRP also outperforms MAODV, resulting in higher packet delivery ratio and lower latency. This is true even though our implementation is based on an improved version of MAODV that pro-actively maintains the single, shared multicast tree. For 1 or 2 senders, as many as 99% or more of the packets are delivered, even in high mobility scenarios. The PDR drops for 5 senders to around 97% and is significantly lower at about 93% or so for 10 senders. There is also a noticeable increase in packet

latency for the 10 senders scenarios. Mobility seems to have little impact on overall performance, with PDR and latency roughly similar across comparable scenarios.

In terms of network efficiency, ODMRP has a lower packet send ratio (PSR) than MAODV or the unicast protocols. In general, the higher the number of multicast receivers, the lower the PSR. The highest PSR value is observed for 10 senders and 10 receivers, at about 5.4 (i.e., for every successful packet delivery, 5.4 packet transmissions occurred at the MAC layer). This can be explained by the need to build and maintain 10 multicast meshes. As the number of receivers increases, these costs are amortized over an increased number of successfully delivered packets, reducing the PSR to 1.4 for 50 receivers. For a single multicast sender, the PSR ranges from 2.7 (10 multicast receivers) to 0.88 (50 multicast receivers).

***Table 9: PDR and latency for ADMR, 1 m/s maximum speed***

| | 1 Sender | | 2 Senders | | 5 Senders | | 10 Senders | |
|---|---|---|---|---|---|---|---|---|
| | PDR | Latency | PDR | Latency | PDR | Latency | PDR | Latency |
| **10 Receivers** | 0.979 | 0.045 | 0.975 | 0.048 | 0.966 | 0.046 | 0.957 | 0.045 |
| **20 Receivers** | 0.977 | 0.052 | 0.981 | 0.048 | 0.969 | 0.052 | 0.963 | 0.047 |
| **30 Receivers** | 0.980 | 0.048 | 0.981 | 0.046 | 0.975 | 0.047 | 0.968 | 0.048 |
| **40 Receivers** | 0.983 | 0.050 | 0.982 | 0.042 | 0.976 | 0.045 | 0.971 | 0.044 |
| **50 Receivers** | 0.983 | 0.057 | 0.986 | 0.040 | 0.979 | 0.047 | 0.973 | 0.047 |

***Table 10: PDR and latency for ADMR, 20 m/s maximum speed***

| | 1 Sender | | 2 Senders | | 5 Senders | | 10 Senders | |
|---|---|---|---|---|---|---|---|---|
| | PDR | Latency | PDR | Latency | PDR | Latency | PDR | Latency |
| **10 Receivers** | 0.957 | 0.039 | 0.956 | 0.038 | 0.946 | 0.041 | 0.930 | 0.043 |
| **20 Receivers** | 0.968 | 0.040 | 0.965 | 0.042 | 0.956 | 0.045 | 0.945 | 0.048 |
| **30 Receivers** | 0.975 | 0.043 | 0.970 | 0.044 | 0.965 | 0.046 | 0.954 | 0.050 |
| **40 Receivers** | 0.977 | 0.044 | 0.973 | 0.048 | 0.969 | 0.048 | 0.962 | 0.051 |
| **50 Receivers** | 0.981 | 0.042 | 0.978 | 0.044 | 0.973 | 0.048 | 0.966 | 0.051 |

AMDR performs consistently better than MAODV. For few multicast senders, it performs worse than ODMRP, and it also appears to be more sensitive to the mobility rate in the network. It does, however, perform better for the 10 multicast senders scenarios, under both low and high mobility. Overall, it is the most stable protocol, giving very consistent performance across all scenarios. Under low mobility, for example, the PDR ranges from 95.7% to 98.6%, with packet latencies varying from 42 to 57 milliseconds. Under high mobility, PDR ranges from 93.0% to 98.1%, and packet latency varies from 38 to 51 milliseconds.

This consistent performance is also shown by the network efficiency metrics. Similar to all multicast protocols, the PSR is reduced as the number of multicast receivers increases. This is mostly due to two reasons: since packets are broadcast at the MAC layer, a higher number of multicast receivers increases the chances that this packet broadcast is received by multiple receivers. Also, protocol overheads are amortized over an increased number of packet deliveries with a higher number of receivers, in particular for protocols such ODMR and MAODV, where certain protocol operations result in the flooding of control packets to ALL nodes in the network, independent of the number of multicast receivers.

ADMR avoids such periodic floods, and consequently its PSR varies to a much smaller degree: from 1.17 to 0.35 under low mobility and from 1.8 to 0.46 for high mobility scenarios.

Multicast protocols improve packet delivery ratio and reduce packet latency, compared to solutions based on unicast routing protocols. This is true for all multicast protocols studied here, clearly indicating that multicasting achieves the stated objectives of increased performance at lower usage of network resources.

The results also show that there are significant differences in the performance of the multicast protocols. Overall, MAODV performed the worst. ODMRP and ADMR have their relative strengths and weaknesses. ODMRP achieves higher packet delivery ratios and lower packet latencies for scenarios with relatively few (1, 2, or 5) multicast senders. ADMR performs better in scenarios with many (i.e., 10) multicast senders. In addition, ADMR performs more consistently across the range of scenarios, and requires less network resources (lower PSR).

### 4.4.3 Broadcast protocols

The last two protocols implement broadcasting: all data packets are delivered to all nodes. We applied these protocols to multicast scenarios (i.e., only a subset of nodes is interested in this data), the metrics reported here are based on the protocol performance with respect to the identified multicast receivers. The overhead caused by the delivery of data packets to nodes that are not multicast receivers is captured by the network efficiency metrics.

***Table 11: PDR and latency for FLOOD, 1 m/s maximum speed***

| | 1 Sender | | 2 Senders | | 5 Senders | | 10 Senders | |
|---|---|---|---|---|---|---|---|---|
| | PDR | Latency | PDR | Latency | PDR | Latency | PDR | Latency |
| **10 Receivers** | 0.998 | 0.023 | 0.991 | 0.027 | 0.956 | 0.051 | 0.828 | 2.084 |
| **20 Receivers** | 0.998 | 0.025 | 0.991 | 0.029 | 0.956 | 0.052 | 0.827 | 2.099 |
| **30 Receivers** | 0.996 | 0.025 | 0.989 | 0.029 | 0.954 | 0.052 | 0.826 | 2.091 |
| **40 Receivers** | 0.996 | 0.026 | 0.989 | 0.029 | 0.954 | 0.052 | 0.826 | 2.078 |
| **50 Receivers** | 0.996 | 0.025 | 0.990 | 0.028 | 0.955 | 0.051 | 0.831 | 2.041 |

***Table 12: PDR and latency for FLOOD, 20 m/s maximum speed***

| | 1 Sender | | 2 Senders | | 5 Senders | | 10 Senders | |
|---|---|---|---|---|---|---|---|---|
| | PDR | Latency | PDR | Latency | PDR | Latency | PDR | Latency |
| **10 Receivers** | 0.999 | 0.023 | 0.993 | 0.029 | 0.965 | 0.053 | 0.815 | 1.684 |
| **20 Receivers** | 0.999 | 0.023 | 0.993 | 0.028 | 0.965 | 0.052 | 0.815 | 1.683 |
| **30 Receivers** | 0.999 | 0.023 | 0.993 | 0.029 | 0.965 | 0.052 | 0.815 | 1.687 |
| **40 Receivers** | 0.999 | 0.023 | 0.993 | 0.028 | 0.965 | 0.052 | 0.815 | 1.686 |
| **50 Receivers** | 0.999 | 0.022 | 0.993 | 0.028 | 0.965 | 0.051 | 0.818 | 1.652 |

Under both low and high mobility, the packet delivery ratio stays high (at or above 99%) for one or two multicast senders, dropping to 95.5% to 96.5% for 5 multicast senders and to 81.5% to 82.7% for 10 multicast senders. The packet latencies are small (a few tens of

milliseconds) for most scenarios, but increase drastically for the 10 multicast senders scenarios. The number of multicast receivers, as expected, does not impact the performance metrics.

Some packet losses can be explained by transmission collisions. However, the protocol implementation takes great care to avoid such collisions, randomly jittering re-broadcasts by 10 ms. Also, the underlying MAC protocol is based on "carrier sense" (i.e., listen to the media and apply random backoff when media is busy), reducing the chances of collision errors even further. In addition, a receiver can receive a specific data packet over multiple different "paths". Loosing the flooded data packet over all such paths due to collisions is rare indeed, as shown by the high packet delivery ratios for 1 and 2 senders. However, further exploring the collected traces, it appears that with 5 or 10 senders, the network starts to experience congestion. The MAC protocol, which is capable of buffering up to 50 packets being passed down from the network layer (where FLOOD is implemented), starts dropping a significant number of packets due to queue overflow.

The network efficiency is largely independent of the number of multicast senders and mobility rate. In FLOOD, every packet is transmitted up to 50 times. Therefore, the PSR is 5 (50 packet transmissions for 10 packet receptions) for 10 multicast receivers and 1 (50 packet transmissions for 50 packet receptions) for 50 multicast receivers. This is approximately true for all multicast sender scenarios. Even when the packet delivery ratio falls to significantly below 100% in the 10 multicast sender cases, the total number of packet transmissions at the MAC layer falls proportionally.

***Table 13: PDR and latency for BCAST (100 ms), 1 m/s maximum speed***

| | 1 Sender | | 2 Senders | | 5 Senders | | 10 Senders | |
|---|---|---|---|---|---|---|---|---|
| | PDR | Latency | PDR | Latency | PDR | Latency | PDR | Latency |
| **10 Receivers** | 0.997 | 0.116 | 0.996 | 0.108 | 0.987 | 0.118 | 0.970 | 0.126 |
| **20 Receivers** | 0.997 | 0.119 | 0.996 | 0.112 | 0.987 | 0.121 | 0.970 | 0.129 |
| **30 Receivers** | 0.995 | 0.121 | 0.994 | 0.113 | 0.985 | 0.121 | 0.968 | 0.129 |
| **40 Receivers** | 0.995 | 0.121 | 0.994 | 0.113 | 0.984 | 0.121 | 0.967 | 0.129 |
| **50 Receivers** | 0.995 | 0.118 | 0.994 | 0.110 | 0.985 | 0.118 | 0.968 | 0.126 |

***Table 14: PDR and latency for BCAST (100 ms), 20 m/s maximum speed***

| | 1 Sender | | 2 Senders | | 5 Senders | | 10 Senders | |
|---|---|---|---|---|---|---|---|---|
| | PDR | Latency | PDR | Latency | PDR | Latency | PDR | Latency |
| **10 Receivers** | 0.993 | 0.104 | 0.992 | 0.108 | 0.986 | 0.114 | 0.968 | 0.126 |
| **20 Receivers** | 0.992 | 0.103 | 0.991 | 0.107 | 0.986 | 0.114 | 0.968 | 0.127 |
| **30 Receivers** | 0.993 | 0.103 | 0.992 | 0.107 | 0.986 | 0.113 | 0.968 | 0.127 |
| **40 Receivers** | 0.993 | 0.103 | 0.992 | 0.107 | 0.986 | 0.113 | 0.968 | 0.126 |
| **50 Receivers** | 0.993 | 0.101 | 0.992 | 0.105 | 0.986 | 0.110 | 0.969 | 0.124 |

When scaling the packet delay factor by 100 ms, BCAST achieves similar performance to FLOOD for one or two multicast sender, but with significantly increased PDR for 5 and 10 multicast senders. For 1 multicast sender, the PDR is slightly below the PDR in FLOOD, in particular for high mobility scenarios. This is due to the fact that BCAST has less redundancy, dynamically selecting only a subset of nodes to re-broadcast a packet.

This shows in the packet send ratio as well: PSR ranges from 2.5 to 0.5 for 10 and 50 multicast receivers, respectively. The resulting lower network traffic is beneficial in the 2 and 5 senders cases, where fewer collisions occur at the MAC layer, resulting in improved PDRs, compared to FLOOD. And scenarios with 10 multicast senders benefit the most, with PDRs of about 97%. Overall, the protocol performance is very consistent across all scenarios, with PDR ranging from 96.7% to 99.7% and packet latency ranging from 104 ms to 129 ms.

Packet latency is rather high, consistently above 100 ms. This is a consequence of the implementation, where packet transmission is delayed by an amount of time randomly chosen from [0, MAX/N * 100 ms], as explained above. Running experiments with a smaller scaling factor, such as 10 ms, shows that a large scaling factor is beneficial, achieving high packet delivery ratio at the expense of some latency. With a 10 ms scaling factor, PDR for the 1-sender cases is around 99.6% to 99.7%, dropping to 98.3% to 98.6% for the 2 senders cases. It drops even more drastically to 91% to 92% for the 5 senders cases, and to 69% for the 10 senders cases. Except for the 10 senders cases, though, packet latency is reduced to about 30 ms in all 1 and 2 sender scenarios, increasing to about 75 ms in the 5 senders cases. All in all, however, for BCAST to perform well, compared to FLOOD, a relatively large delay factor is beneficial.

Broadcasting appears to be an attractive way to achieve good multicast performance. Both protocols achieve high packet delivery ratios for most scenarios; protocol performance only starts to suffer for 10 multicast senders. FLOOD is a simple protocol, and BCAST shows that the same or even better protocol performance may be achieved with significantly less overhead. The performance of both protocols is insensitive to the number of multicast receivers and the mobility rate, and seems primarily determined by the overall network traffic (which scales linearly with the number of multicast senders in our setup).

## 4.5 Conclusions

There are a number of alternatives when delivering data from one or a few senders to a group of receivers: setting up dedicated unicast connections from each sender to each receiver, employing a multicast protocol, and broadcasting the packet to every node. The experiments reported here show that reducing the multicast problem to an n-fold unicast case is the worst solution: except for scenarios with only one or two senders and a small number of receivers, packet delivery ratio is low and packet latencies are high. This is mostly due to two reasons:

1. For scenarios with N senders and M receivers, NxM unicast connections have to be discovered and maintained by the underlying unicast routing algorithm, introducing a substantial protocol overhead.

2. The replication of data packets on the sender side for each of the M receivers results in high traffic loads for the MANET. Both multicast and broadcast protocols exploit the inherent broadcast nature of the underlying medium to significantly reduce this traffic load.

Any of the three multicast protocols we studied improve the multicast performance. Among them, MAODV had the poorest performance. Based on our analysis, this is due to the shared multicast tree, for two reasons:

1. The tree is based on hard state, requiring explicit control messages to maintain it, in particular under high mobility scenarios. ODMRP and AMDR, by contrast, maintain their multicast distribution structure in soft state.

2. With all data flowing through a shared tree, the queues along interior tree nodes are more likely to overflow, even for a small number of multicast senders.

ADMR and ODMRP show better performance. ODMRP in particular works well for relatively few multicast senders; ADMR works well for 10 multicast senders scenarios, has the most consistent performance among all multicast protocols across all scenarios, and is very network efficient. The broadcasting protocols work very well in most scenarios, and are more robust with respect to number of multicast receivers and mobility rate. BCAST improves on FLOOD in most cases and is more network efficient, indicating that it pays off to explore more complicated broadcast protocols.

None of the broadcast/multicast protocols outperforms the other protocols in all scenarios. The following tables summarize the best protocol for each scenario, ranked by PDR (with packet latency as tie-breaker). Any ranking, by necessity, will have to weigh different performance metrics such as PDR or latency. We chose PDR as the more significant metric for the following reasons:

1. First, the two metrics are not mutually exclusive. The above results show that significant drops in packet delivery ratio are usually accompanied by large increases in packet latency, indicating that the network starts to experience congestion. So certainly protocols can exhibit simultaneously poor performance under both metrics.

2. As long as packet latency is bound by a relatively small number, we assume that further improvements in latency do not really contribute to the overall user satisfaction. For example, the ITU recommends that VoIP applications should not experience latencies beyond 400 ms (roundtrip) for voice service. All entries in the tables below have one-way latencies well below 200 ms, meeting the roundtrip upper bound.

3. Many applications are sensitive to PDR. For example, data download times directly benefit from improvements in PDR. Furthermore, the throughput of a TCP-like data stream will suffer with every packet loss, so even small improvements in PDR can yield significant improvements in application performance and user satisfaction.

4. Finally, we are interested in researching reliable multicasting protocols. As starting point, we would therefore like to identify those protocols that already provide relatively high packet delivery ratios.

***Table 15: Protocol ranking, 1 m/s maximum speed***

| | 1 Sender | | 2 Senders | | 5 Senders | | 10 Senders | |
|---|---|---|---|---|---|---|---|---|
| | PDR | Latency | PDR | Latency | PDR | Latency | PDR | Latency |
| **10 Receivers** | FLOOD 0.998 | 0.023 | AODV 0.996 | 0.024 | BCAST 0.987 | 0.118 | BCAST 0.970 | 0.126 |
| **20 Receivers** | FLOOD 0.998 | 0.025 | BCAST 0.996 | 0.112 | BCAST 0.987 | 0.121 | BCAST 0.970 | 0.129 |
| **30 Receivers** | FLOOD 0.996 | 0.025 | BCAST 0.994 | 0.113 | BCAST 0.985 | 0.121 | BCAST 0.968 | 0.129 |
| **40 Receivers** | FLOOD 0.996 | 0.026 | BCAST 0.994 | 0.113 | BCAST 0.984 | 0.121 | ADMR 0.971 | 0.044 |
| **50 Receivers** | FLOOD 0.996 | 0.025 | BCAST 0.994 | 0.110 | BCAST 0.985 | 0.118 | ADMR 0.973 | 0.047 |

***Table 16: Protocol ranking, 20 m/s maximum speed***

| | 1 Sender | | 2 Senders | | 5 Senders | | 10 Senders | |
|---|---|---|---|---|---|---|---|---|
| | PDR | Latency | PDR | Latency | PDR | Latency | PDR | Latency |
| **10 Receivers** | FLOOD 0.999 | 0.023 | FLOOD 0.993 | 0.029 | BCAST 0.986 | 0.114 | BCAST 0.968 | 0.126 |
| **20 Receivers** | FLOOD 0.999 | 0.023 | FLOOD 0.993 | 0.028 | BCAST 0.986 | 0.114 | BCAST 0.968 | 0.127 |
| **30 Receivers** | FLOOD 0.999 | 0.023 | ODMRP 0.994 | 0.012 | BCAST 0.986 | 0.113 | BCAST 0.968 | 0.127 |
| **40 Receivers** | FLOOD 0.999 | 0.023 | ODMRP 0.994 | 0.012 | BCAST 0.986 | 0.113 | BCAST 0.968 | 0.126 |
| **50 Receivers** | FLOOD 0.999 | 0.022 | FLOOD 0.993 | 0.028 | BCAST 0.986 | 0.110 | BCAST 0.969 | 0.124 |

Overall, the following major trends can be identified:

- Broadcast protocols work rather well in the scenarios we studied. BCAST and FLOOD work almost always as good as or better than other protocols, though sometimes impose higher packet latency.

- For a single multicast sender, FLOOD is the obvious choice; for increasing number of multicast senders, BCAST has the edge over FLOOD

- ADMR performs very well in the presence of many multicast senders, and is indeed the optimal choice in two scenarios under low mobility, with BCAST being runner-up. All other protocols perform poorly in these scenarios.

- The choice of an optimal multicasting solution is largely independent of the mobility rate. This is true for the 1 and 5 multicast senders scenarios, but also largely true in the 2 multicast senders scenarios. In the latter case, it is often the tie-breaking criterion that results in the selection of a specific protocol in Tables 15 and 16. In all cases, BCAST achieves the same maximal PDR, but at higher packet latency. For the 10 senders scenarios, BCAST and ADMR are the only two protocols to provide consistently high performance, with BCAST's PDR being higher in the vast majority of cases.

We can also rank the protocols by average PDR across all scenarios, assuming all scenarios are equally likely. Ideally, we would like to achieve 100%, perfect packet delivery. However, this is an unreachable ideal, since network partitions do occur in the simulations (i.e., senders and receivers are in disjoint parts of the network, unable to communicate with each other). A more realistic upper bound is indicated by the results of Tables 15 and 16. Averaging across all scenarios, the PDR for an ideal multicast routing protocol (performing as well or better than the best protocol listed in these tables for the various scenarios) is at least 98.6775%, with an average packet latency of 0.071775 seconds.

***Table 17: Protocol ranking based on average performance***

| | Packet Delivery Ratio (%) | Packet Latency (sec) |
|---|---|---|
| "Ideal" Protocol | 98.6775 | 0.071775 |
| BCAST | 98.5408 | 0.115765 |
| ODMR | 96.9965 | 0.020807 |
| ADMR | 96.9200 | 0.045980 |
| FLOOD | 94.2792 | 0.495640 |
| MAODV | 72.9593 | 0.446661 |
| AODV | 48.3833 | 1.383730 |
| DSR | 25.7493 | 2.320680 |

The results, summarized in Table 17, show that BCAST comes close to the "ideal" protocol in PDR, at the cost of higher packet latency. ODMR and AMDR are, on average, performing better than FLOOD, which suffers from its relatively poor performance in the 10 senders scenarios. The unicast protocols are clearly the worst choices, with an average PDR below 50% and average packet latencies in excess of 1 second.

Both the results published in the literature and our own experiments show that achieving a high packet delivery ratio is hard. Even in the absence of network congestion, the major cause of packet loss in most of our experiments, multicast receivers will not receive all packets. We therefore need to explore ways to

1. Increase the packet delivery ratio for low traffic levels.

2. Increase the network capacity to support more high-traffic scenarios.

The second objective will require work along two lines: reducing the routing protocol overhead and reducing the MAC protocol overhead/increasing the link capacity, which is discussed towards the end of the report. However, we first explore the first approach, increasing the packet delivery ratio. Since BCAST gave us, overall, high packet delivery ratio with relatively low overhead, we will use it as a starting point.

# 5. Design alternatives

A reliable multicast protocol has to address a number of key issues. The two overriding and orthogonal issues are what mechanism should be used to ensure reliability and whether the protocol is implemented as a transport layer protocol or as a network layer protocol. Within each of these two issues, specific sub-issues need to be addressed. Finally, flow control and security are considered important high-level issues as well. This section discusses the alternatives and justifies our high-level design decisions.

## 5.1. Reliability mechanism

The previous section showed that current multicast approaches do not deliver a sufficiently high number of data packets to be considered suitable. So we have to assume that data packets will be lost in transmission. To recover from this loss, two (complementary) approaches are possible: the use of forward error correction codes and packet retransmissions.

Protocols based on forward error correction (FEC) codes take a data packet, split it up, and add additional, redundant information. These sub-packets are then transmitted to all multicast receivers. Assuming that these smaller packets are lost independently, a receiver can re-assemble the original data packet as long as a sufficient subset of smaller packets is received. In general, the more redundant information the sender adds, the more packet losses can be tolerated. Example protocols based on this idea are described in [Chumchu 2002, Shu 2002].

Protocols based on packet retransmission involve some form of automatic repeat request (ARQ) scheme. A receiver has to be able to identify that it missed a packet transmission, which is typically achieved by assigning packets sent by the same sender a unique and monotonically increasing sequence number. When receivers detect a gap in the packet sequence, they will ask for retransmission of this packet, with retransmission requests either directed to the sender or some intermediate node that is known or suspected of having stored a copy of previously transmitted packets. Protocols based on this idea are described in [Gopalsamy 2002, Jiang 2002, Kuri 2001, Omar 2002]

FEC-based approaches are attractive when the communication links are unidirectional, since they do not require any feedback to be sent back to the sender. They also, in general, have lower recovery latencies than retransmission-based protocols. However, most reliable multicast protocols are based on the idea of packet retransmissions, for the following reasons:

- FEC-based protocols need to know the worst case packet loss to generate enough redundant data to ensure correct packet delivery.

- FEC-based protocols typically increase network traffic, even when packet loss rates are low.

Since the use of forward error correction codes alone is often insufficient, and most wireless media allow for bi-directional communication, packet retransmission is often employed even in protocols that are primarily based on the use of FEC, for example [Chumchu 2002]. Therefore, we will only explore the use of packet retransmission to achieve reliable multicasting.

A number of specific issues arise when considering packet retransmission. First, should the receivers explicitly request the retransmission of missed packets in the form of negative acknowledgements (NACKs) or should the receiver indirectly communicate this information by only sending positive acknowledgements (ACKs), similar to TCP's acknowledgement mechanism? In a multicast scenario with one or a few senders and many receivers, overwhelming nodes with status information from many receivers (control information implosion) is a serious concern. Under the assumption that losses occur less often than packet receptions, NACKs seem to be the preferred choice.

Second, whom should the receiver contact for retransmission? Ultimately, it could be the original sender, but the sender can be far away, leading to a long recovery time. Many protocols try to speed up the recovery latency and distributed the load by having other nodes buffer data packets and retransmit them on demand. This works particularly well in cellular networks, where base stations/access points are often used for this purpose, servicing all mobile nodes in their area [Anastasi 2001, Kuri 2001, Omar 2002]. In a MANET, no such special nodes exist, so either potentially every node provides this functionality or the multicast protocol provides mechanisms to identify alternative sources of missed packets.

Third, every node that will retransmit packets on demand will have to buffer them. As mentioned above, nodes only have a finite amount of buffer space, so buffer management becomes important. Ideally, the multicast protocol needs to allow such nodes to determine when all receivers received a given packet, at which point they can discard the packet from their buffer. ACKs (rather than NACKs) do provide this information, which is utilized by some multicast protocols. However, this buffer management also introduces another subtle issue: as introduced above, senders send data packets to an IP multicast address, not knowing and usually not caring as to which senders are currently joined. However, the buffer management strategy described above implied that a node/sender knows when ALL receivers received the data packet, i.e., such nodes know the current group membership information. Many reliable multicast protocols for fixed networks therefore provide reliable group membership protocols as part of the package that allow senders to have an accurate knowledge about the current multicast group status. Since, in a MANET, nodes may be disconnected from the others for unpredictable amounts of time, protocols that do not require accurate knowledge of the group status are preferred.

## 5.2. Transport layer vs. routing layer

Another major issue concerns whether the reliable protocol should be implemented at the transport layer or at the network layer (i.e., as part of the multicast routing protocol). As stated in [Obraczka 1998], more recent multicast protocols tend to follow a design model referred to as "application-layer framing": they are designed for specific applications, rather than as general-purpose protocols. As such, this concept does not necessarily imply that specific functionality is implemented at the transport or application layer. But as more protocols are developed, there is some benefit to separating functionality into multiple layers, providing more generic support at lower layers and more specific functionality at higher layers.

Reliable multicast protocols exemplify such an application-specific approach, and based on the preceding logic therefore should exist higher in the protocol stack than the general-purpose IP routing protocols. This also is one approach currently favored by the IETF who chartered a Working Group on Reliable Multicast Transport Protocols (http://www.ietf.org/html.charters/rmt-charter.html). Examples of reliable multicast protocols designed for specific applications, exploiting application-specific features, are described in [Adamson 2002, Zhuang 2001].

Separating the functionality into different protocol layers has quite a history within the IETF, the most obvious example being the split between IP and TCP. IP provides best-effort connectivity between hosts, TCP provides true end-to-end communication between applications running on these hosts and provides the abstraction of a reliable data stream over a best-effort IP network. To achieve this, TCP adds port numbers to the IP addresses to uniquely identify specific senders and receivers on each host, and an ARQ mechanism to retransmit lost IP packets.

Separating routing and reliability into two layers has a number of advantages. It allows the reliable multicast protocol to be independent of the underlying multicast routing protocol. This in turn allows the transport layer protocol to benefit from any improvement of the routing protocol. Furthermore, changes to the routing protocol will be confined to the routers, not requiring changes to the end hosts. However, in a MANET, it is questionable whether these advantages still hold. Since all nodes are end hosts and routers, changes to either the reliable multicast protocol or the multicast routing protocol will result in updates to all hosts. In addition, as the experience with TCP shows, a true end-to-end transport layer protocol performs rather poorly in a wireless environment. Following the end-to-end argument of Salzer et al. [Salzer 1984], pushing functionality down the protocol stack often results in better performance. To efficiently deal with the characteristics of the wireless links and the mobility of all nodes, per-hop actions may be preferred. For example, for faster recovery from packet loss, intermediate nodes in the multicast distribution structure may be asked to retransmit the packet. A transport layer protocol is only defined between the communicating peers, so packet retransmissions will have to be requested from the sender in this case. Since a sender is potentially multiple hops away, this can easily result in either the retransmission request or the retransmitted packet to be lost again. Furthermore, even if the retransmitted packet is successfully received, this may take a relatively long time. Therefore, we will implement the reliability mechanism together with the routing protocol at the network layer.

## 5.3. Flow control and security

Finally, the issues of flow control and security need to be addressed. As stated in [RFC 2357], reliable multicast applications may experience a potential explosion of complex patterns of control traffic; therefore the design of congestion control mechanism for reliable multicast for large multicast groups is essential. [Li 2002] discusses in depth some of the challenges and approaches to address multicast flow and congestion control. In this work, we assume that multicast groups, on average, are not really huge or geographically widespread (i.e., throughout the Internet). Rather, a limited number of nodes within a geographically confined area only will communicate via multicast, so we will ignore, for the time being, this issue. Should the multicast group span multiple ad-hoc networks, we expect that the connectivity between these networks will be provided by a fixed infrastructure. As discussed in [Ding 2002], such a combination of MANETs and fixed networks will require the use of multiple multicast protocols: one in the fixed network and a different one in the MANETs. These protocols will be connected by special multicast gateways, with flow and congestion control being the responsibility of the fixed network multicast protocol.

Another flow-control related aspect is the very limited capacity of a MANET. As reported in the literature and demonstrated by the experiments discussed in the previous section, the effective bandwidth available to a single node decreases rapidly with the number of retransmissions per packet [Li 2001]. This is a particular concern for multicast and broadcast schemes, where a single packet may need to be re-transmitted many times. In these situations, it becomes rather easy to overload/congest the MANET. As reported in [Williams 2002], all broadcast schemes have a

breakpoint: as the severity of the wireless environment increases (in terms of mobility, traffic load, and number of nodes), the packet delivery ratio of a protocol will deteriorate rapidly beyond a specific point. In this work, we will explore the performance of various multicast schemes under different scenarios to identify this breakpoint. For some protocols, we may be able to push the performance by judiciously choosing the appropriate protocol overhead (interval of hello messages, piggybacking control information on data packets, etc.). But ultimately we can expect that all protocols will break when exposed to a very severe (i.e., causing permanent network congestion) environment. For the time being, we will not explore how to control the sender rate to achieve or guarantee reliable packet delivery.

Security is increasingly becoming a concern for all IETF protocols. As listed in [RFC 2357], the following security and privacy concerns need to be addressed: which parties (senders, routers, receivers, retransmission sources, etc.) must be trusted in order to ensure secure operation and privacy of the transmitted data? Does the protocol allow a group of receivers to determine whether they all received the same data (i.e., the Byzantine General Agreement problem)? Are there limitations on the retransmission mechanism to prevent it from being abused to flood network links with excessive traffic (i.e., denial of service attacks)? These questions will be addressed in the second part of the study.

# 6. Reliable BCAST

## 6.1. Related work

A number of researchers have proposed reliable multicast protocols for fixed networks (a detailed survey can be found in [Li 2002]) and cellular networks [Anastasi 2001, Kuri 2001, Omar 2002]. The latter protocols typically assign a special role to the base stations/access points, and are therefore not applicable in a MANET environment. Relatively few papers have addressed the issue of reliable multicasting in a MANET. [Tang 2002] proposes RALM (Reliable Adaptive Lightweight Multicast), a multicast transport layer protocol that achieves relatively high packet delivery by throttling traffic using a window mechanism based on congestion experienced by a feedback receiver. [Gopalsamy 2002] describes RMA, a Reliable Multicast Protocol for MANETs. The protocol is based on the assumption that senders know the identities of all receivers and achieves reliability by explicit ACKs from all receivers. The protocol achieves high packet delivery ratio with a data packet overhead similar to MAODV. Finally, [Shu 2002] discusses how to assure packet delivery in MANETs using error correction codes: packets are encoded and split into smaller pieces, which are transmitted independently to the receiver. Assuming a certain threshold of pieces arrive at a receiver, the packet can be re-assembled. No performance evaluation is provided in the paper, however.

Our approach is different from these attempts: we increase packet delivery at the routing layer, unlike [Tang 2002], which described work at the transport layer, for reasons discussed above. Our protocol does not require the senders to know the identity of all senders, unlike [Gopalsamy 2002], and it is based on packet retransmission and not forward error correction codes, as in [Shu 2002].

## 6.2 The protocol

The basic mechanism to improve the packet delivery ratio in BCAST is fairly straightforward: every node buffers the last X packets. X can be any arbitrary number, to keep the memory requirement at each node low, we set X to a small number (10 packets) in our code. The buffer is implemented in round-robin fashion, storing the last X unique packets a node received (from the same or a different sender, in order or out of order).

When a node receives a packet with sequence number S from source node SRC, it checks whether it also received packet S-1 from the same source. If not, a node issues a 1-hop broadcast to the neighbors, asking for retransmission of this packet (the NACK message). Each neighbor, upon receiving the NACK packet, checks its local cache and retransmits the packet (assuming it has it in its local buffer). To reduce collisions, the NACKs and the packet retransmissions are jittered randomly by 10 milliseconds. In addition, NACKs have a timeout mechanism associated with them, so even if a NACK or retransmission is lost, packets can be recovered. NACKs are re-issued up to a certain maximum number of attempts (up to 3 times in our implementation).

To reduce network traffic, nodes with pending packet retransmissions will cancel their retransmission if they overhear another node X re-broadcasting this packet. This is based on the assumption that the requesting node will receive this packet as well, satisfying the NACK. This is arguably not guaranteed to be the case: node X could be out of reach of the requesting node,

broadcasting packet N for other reasons. However, with multiple NACK attempts (spaced apart multiple seconds), eventually only nodes that received a NACK will attempt to re-transmit a packet. Since they received the NACK, and packets are retransmitted with little additional delay, it is reasonable to assume that the requesting node, in turn, will receive their transmission.

If a sequence of packets is lost, our NACK mechanism recovers from this by backtracking. Assume that packets 3-6 from source S are lost. When a node receives packet 7 from S, it will trigger a NACK for packet 6. Once packet 6 is received, a NACK for packet 5 is triggered, etc. Finally, upon receiving a re-transmitted packet 3, the node determines that it already received packet 2 and the backtracking will stop. Note that this backtracking will also terminate once a packet cannot be recovered. For example, if the node is unable to recover packet 4, it will not check for receipt of packet 3, and therefore not trigger a NACK for packet 3. All else being equal, this is not unreasonable: since nodes buffer the most recent packets, being unable to retrieve packet 4 from a neighbor is a strong indication that older packets (such as packet 3) may also not be available from those neighbors.

With this NACK based scheme, the packet delivery ratio can be improved. For example, in the 1 sender/10 receiver scenarios, the packet delivery ratio increases from 99.653% to 99.822% under low mobility. In 8 out of 10 simulation scenarios, the packet delivery ratio was a perfect 100%. However, there are still sources of packet loss that a NACK-based scheme cannot completely avoid: problems due to the NACK mechanism itself and long-lived network partitions.

1. Persistent loss of NACK/retransmission

   Due to collisions, the NACK or packet re-broadcast may be lost. In this case, the mechanism fails. We improve the reliability of the NACK mechanism by using a timer and re-issuing NACKs if needed. Since, for reasons discussed below, not every packet is recoverable, we keep the number of NACK retransmission attempts to a small number (3) to avoid inducing a high load on the network.

2. Network partitions

   Due to the dynamic nature of MANETs, the network may partition for a lengthy period of time. The two scenarios for the 1 sender/10 receiver cases that do not achieve 100% packet delivery ratio exemplify this situation, although they also highlight two important variations. In these two scenario, node 0 is the multicast sender and nodes 40 through 49 are the multicast receivers. In the first scenario, all receivers receive all packets except for node 44, which misses out on packets 13 to 246. Closer examination of this scenario reveals that up to time 36.55 seconds, node 44 is within communication range of node 36, receiving packets 1 through 12. At 36.55 seconds, the two nodes have moved more than 250 meters (the communication range) apart, node 44 becomes disconnected from the rest of the network (together with the non-receiver nodes 4, 20, and 30). At time 156.6 seconds, node 20 has moved sufficiently close to node 7 to establish a link, node 44 is within range of node 20, and packets are delivered again to this receiver. Node 44 recovers, by backtracking, from a number of missed packets, but with nodes only buffering up to the 10 most recent data packets, the recovery is limited by the buffer size and the number of multicast senders. In essence, under low traffic and with a single multicast sender, 10 packets allow to bridge network partitions of up to 5 seconds (2 packets/second traffic rate). To recover from a network partition of 120 seconds, the nodes would have to buffer approximately 240 or more packets, more if there was more than one multicast sender.

Even if nodes were to buffer many packets, this does not guarantee 100% reliable data delivery. In the second scenario, the partition occurs towards the end of the simulation, at time 865.2 seconds. Receiver 42 is communicating with neighbors, in particular node 30, until time 865.2 seconds. Exploring the node positions, at that time the two nodes are over 250 meters apart, i.e., out of their communication range. This partition remains until the end of the simulation, so node 42 missed out on the last 81 packets. However, in the absence of a new packet with higher sequence number, this situation is indistinguishable from the case of the sender having stopped transmitting, so no NACKs are triggered (nor would this help, since the node remains disconnected from the network).

We conducted a first set of simulations based on the mechanisms described above: every node buffers the 10 most recent packets, NACKs are re-issued if we fail to receive a packet for up to three times, with a NACK timeout value of 1 second. Due to the above reasons, we did not expect a perfect 100% packet delivery ratio, however, we expected an improved packet delivery rate across all scenarios, in particular for the 10 sender cases, where less than 97% of packets were delivered.

The results for small numbers of multicast senders confirmed our expectations: packet delivery ratios went up to over 99.6% for 1 and 2 sender and consistently above 99.2% for five senders under low mobility. For high mobility scenarios, the PDR was 99.9% for 1 sender, 99.8% for two senders, and 99.5% for five senders. These results show a trend we noticed consistently: high mobility scenarios achieve better performance than low mobility scenarios. The explanation, based on the above discussion, is rather straightforward: under high mobility scenarios, potential network partitions may be more frequent, but are also more short-lived, resulting in fewer packet losses (and therefore a greater chance of being recoverable from a limited-size cache). However, the results for the 10 sender scenarios were actually worse then the best-effort protocol, dropping to as low as 70%.

Upon closer examination of the results, we discovered that the NACK mechanism potentially results in many NACKs. In the 10 sender scenarios, close to 225,000 NACKs were issued on average per simulation run. This number is significantly higher than the number of NACKs issued in scenarios with fewer multicast senders: around 75 NACKs were issued on average in the 1 sender cases, around 200 NACKS in the two sender cases, and around 2000 NACKS in the five sender cases. This high number of NACKs (plus the packet retransmission they trigger) substantially adds to the network load, resulting in congestion and packet losses. It is clearly apparent from these numbers that NACKs have to be rate controlled: if the network is busy, aggressively asking for packet retransmissions makes a bad situation only worse.

After exploring a number of options, we settled in the following mechanism to limit the number of NACKs: each node monitors the traffic density by keeping track of when it sends a packet or receives one. This is done in a sliding window of fixed size. Before sending a NACK, the node checks how long ago the earliest packet in that sliding window was send/received. If this period is too short (indicating a high network traffic load), the NACK is suppressed. The size of the sliding window (30) and the minimum required time difference (0.4 secs) are derived at experimentally and balance the need for allowing many NACKs to go ahead with the need to prevent congestion a heavily loaded network. With these parameter settings, few to no NACKs are suppressed in the 1, 2, and 5 multicast sender scenarios, but only 5300 NACKs are transmitted for 10 sender scenarios, reducing the number of NACKs by over 97%.

## 6.3. Reliable BCAST performance

***Table 18: PDR and latency for reliable BCAST, 1 m/s maximum speed***

| | 1 Sender | | 2 Sender | | 5 Sender | | 10 Sender | |
|---|---|---|---|---|---|---|---|---|
| | PDR | Latency | PDR | Latency | PDR | Latency | PDR | Latency |
| **10 Receiver** | 0.998 | 0.117 | 0.998 | 0.109 | 0.995 | 0.126 | 0.972 | 0.130 |
| **20 Receiver** | 0.998 | 0.120 | 0.998 | 0.113 | 0.995 | 0.129 | 0.972 | 0.134 |
| **30 Receiver** | 0.997 | 0.122 | 0.997 | 0.114 | 0.993 | 0.129 | 0.970 | 0.134 |
| **40 Receiver** | 0.996 | 0.123 | 0.996 | 0.114 | 0.992 | 0.129 | 0.970 | 0.134 |
| **50 Receiver** | 0.997 | 0.119 | 0.997 | 0.111 | 0.993 | 0.126 | 0.970 | 0.131 |

***Table 19: PDR and latency for reliable BCAST, 20 m/s maximum speed***

| | 1 Sender | | 2 Sender | | 5 Sender | | 10 Sender | |
|---|---|---|---|---|---|---|---|---|
| | PDR | Latency | PDR | Latency | PDR | Latency | PDR | Latency |
| **10 Receiver** | 0.999 | 0.109 | 0.999 | 0.113 | 0.996 | 0.125 | 0.970 | 0.132 |
| **20 Receiver** | 0.999 | 0.108 | 0.999 | 0.113 | 0.995 | 0.126 | 0.970 | 0.132 |
| **30 Receiver** | 0.999 | 0.108 | 0.999 | 0.113 | 0.996 | 0.122 | 0.970 | 0.132 |
| **40 Receiver** | 0.999 | 0.107 | 0.999 | 0.112 | 0.996 | 0.122 | 0.970 | 0.132 |
| **50 Receiver** | 0.999 | 0.105 | 0.999 | 0.110 | 0.996 | 0.120 | 0.971 | 0.129 |

Tables 18 and 19 summarize the PDR and packet latency under low and high mobility for our reliable BCAST protocol, using NACK rate control. The packet delivery ratio is consistently high (well above 99%) for 1 through 5 multicast senders. As discussed above, higher rates of mobility are good for achieving high packet delivery ratios, since potential network partitions are shorter and can more likely be recovered from with a limited-size cache. For 10 multicast receivers, the packet delivery ratio is significantly lower at 97%, but still improved over the unreliable version by about 0.2%. Under low mobility, packet latency increases only slightly, usually only by about 1 or 2 ms. Under high mobility, average packet latency increases more significantly, by about 5-6 ms for the 1, 2, and 10 multicast sender scenarios, and over 10 ms in the 5 multicast sender scenarios. These increases are in line with the number of packet losses our NACK-based scheme recovers from, which is highest for the 5 multicast sender scenarios, adding to the average packet latency in those scenarios.

## 6.4. Conclusions

This section summarizes our experiences with implementing a reliable multicast protocol. Based on a study of the performance of various alternatives, we selected a broadcast protocol, BCAST, and extended it with a NACK mechanism to increase the packet delivery ratio. As the discussion shows, designing a reliability mechanism to improve packet delivery ratio is non-trivial. Due to the dynamic nature of MANETs, achieving 100% packet delivery is unrealistic, since nodes can and will become partitioned from the multicast senders. In particular under low mobility, such network partitions can exist for lengthy periods of time, recovering from them is therefore not trivial, and would require substantial buffer space at all MANET nodes. It is also probably not desirable from an application perspective: in a VoIP application, for example, a user will assume that the call is disconnected and terminate the application, rather than waiting for 100 seconds to continue the conversation. Similarly, delays of 100 seconds will probably cause a user to terminate interactive downloads and web browsing sessions.

Our protocol achieves consistently over 99% packet delivery ratio under both low and high mobility for a small number of multicast senders. We also noted that higher mobility rates are actually beneficial for achieving high packet delivery ratios, as potential network partitions tend to be more short-lived and therefore can more easily be recovered from. As the number of senders (and therefore the overall traffic load) increases to 10 senders, congestion starts to set in and packet delivery drops to around 97%. Under this high traffic load, the number of NACKs and packet retransmissions has to be carefully controlled to not increase the overall rate of congestive losses. Our protocol therefore implements a mechanism to rate control NACKs based on traffic load.

Future work could progress along a number of lines. First, one could explore ways to fine-tune the protocol parameters, such as the NACK throttle mechanism or "Hello" interval. Ideally, appropriate parameters could be derived automatically, based on observed network traffic and rate of neighborhood change. Second, as discussed above, we could explore ways to increase the network capacity by modifying the MAC protocol. Since this is the more promising approach, we focus on this work. As more traffic can be carried, packet delivery ratios should increase, in particular for scenarios with many multicast senders and/or more data per sender.

# 7. Impact of MAC layer

## 7.1. Introduction

Achieving high packet delivery ratios is challenging. A major source of packet loss is due to network congestion. The primary factor influencing network congestion (besides the offered load) is the network capacity, which is greatly influenced by the data rate of the wireless link and the MAC protocol. The vast majority of simulation studies in NS2 use the IEEE 802.11 MAC layer protocol implemented by the Monarch group at CMU in 1998. This MAC layer is based on a Lucent WaveLan radio with transmission range of 250 meters and a link layer data rate of 2 Mbps. Modern Wireless LAN products, based on 802.11a or 802.11b, provide much higher data rates: up to 11 Mbps for IEEE 802.11b and up to 54 Mbps for IEEE 802.11a.

In addition, a number of papers have shown that MANETs experience significant inter-layer interactions [Ayyagari 2000, Barrett 2002, Fang 2002, Takai 2001]. Interactions between the routing and MAC layer, for example, work in both directions: with directional antennas and power control, routing algorithms are not necessarily constrained to an existing network topology, but can influence node transmission characteristics to achieve end-to-end connectivity. At the same time, simulation studies have shown that MAC layer properties have a significant impact on routing protocol performance, both in absolute and relative terms. The ranking of different routing protocols can be MAC-layer dependent.

As the IEEE 802.11 family of protocols is dominant in the market, we focus on members of this protocol family only. An NS2 MAC layer implementation that can simulate various IEEE 802.11 protocols is provided by INRIA [INRIA 2003, Romdhani 2003]. The main purpose of their work was to explore EDCF, work under study by the IEEE 802.11e working group to support QoS in Wireless LANs [Ni 2002], and to propose enhancements to EDCF in the form of Adaptive EDCF (AEDCF) [Romdhani 2002]. Based on their simulation code, we explore the impact of various MAC protocols on the performance of BCAST and reliable BCAST, the results and conclusions are presented in this report. Certainly, for MAC layers with higher nominal link bandwidth, we expect increased packet delivery ratios and reduced packet latencies.

## 7.2. The MAC layers

Since we could not port the INRIA MAC layer code to our version of NS2 (the changes are spread over numerous files, and there is no documentation available to help in such a port), we ported BCAST and reliable BCAST to the NS2 version distributed by INRIA. In a first step, we confirmed that, under light load (each sender sends 2 data packets per second, each packet has a length of 256 bytes) and with a 2Mbps MAC layer, we obtain roughly the same results as running BCAST in our previous version of the simulator.

The INRIA code and sample scripts contain a number of parameters that can be set to simulate a range of IEEE 802.11 MAC variations. The parameter settings we used for various MAC variants are listed below, focusing only on those that are different (other parameters such as short and long retry limits, are the same for all MAC variants). The first group of parameters sets the MAC layer to operate as IEEE 802.11 MAC, with a link layer data rate of 2 Mbps (consistent with the values in the original MAC implementation):

```
Mac/802_11 set bandwidth_               2Mb
MAC_MIB set RTSThreshold_               0
PHY_MIB set MinimumBandwidth_           1Mb
PHY_MIB set SlotTime_                   0.000020
PHY_MIB set SIFS_                       0.000010
PHY_MIB set CWMin_0                     31
PHY_MIB set CWMax_0                     1023
```

The second set of parameters causes the MAC layer to behave like an IEEE 802.11b MAC, with a link layer data rate of 11 Mbps:

```
Mac/802_11 set bandwidth_               11Mb
MAC_MIB set RTSThreshold_               3000
PHY_MIB set MinimumBandwidth_           6Mb
PHY_MIB set SlotTime_                   0.000020
PHY_MIB set SIFS_                       0.000010
PHY_MIB set CWMin_0                     31
PHY_MIB set CWMax_0                     1050
```

The third set of parameters sets the MAC layer to operate as IEEE 802.11 PHY mode-6, with a data rate of 36 Mbps [Romdhani 2003]. Note that there are other operational modes and data rates, but since the purpose of the study is the explore the impact of different MAC layer protocols on the performance of BCAST, we will use only this one mode.

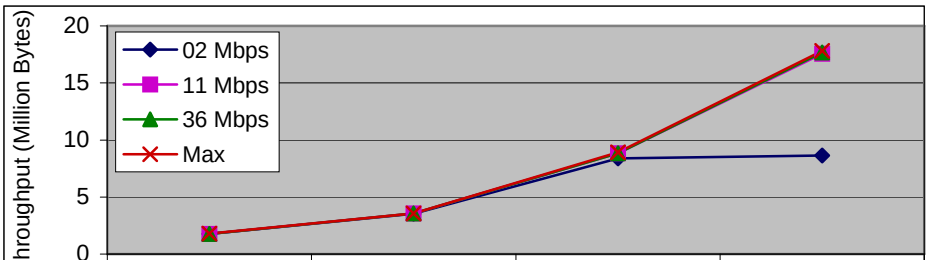
```
Mac/802_11 set bandwidth_               36Mb
MAC_MIB set RTSThreshold_               3000
PHY_MIB set MinimumBandwidth_           6Mb
PHY_MIB set SlotTime_                   0.000009
PHY_MIB set SIFS_                       0.000016
PHY_MIB set CWMin_0                     15
PHY_MIB set CWMax_0                     1050
```

In addition, the INRIA code allows us to specify up to MAX_PLEVELS different traffic classes and to set different initial MIN and MAX congestion windows for each of them. In this work, we do not use this option, treating data and control packets (the periodic HELLO messages) the same. One interesting avenue of future research could be to explore how prioritizing NACK, HELLO, and Data packets would impact the protocol performance.

## 7.3. BCAST performance

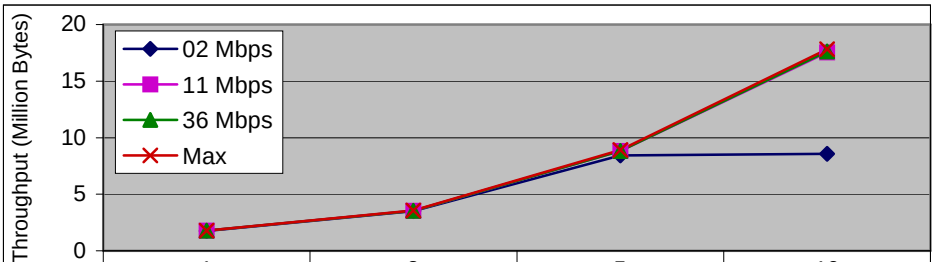
We executed the best-effort version of BCAST with medium and heavy load scenarios over the three MAC layers. Under medium load, each multicast sender transmits 4 packets per second, each packet has a length of 512 bytes. Under heavy load, each multicast sender transmits 8 packets per second, each packet has a length of 1024 bytes. Since the protocol performance is independent of the number of multicast receivers, we average the results for 50 runs (10 repetitions each for scenarios with 10, 20, 30, 40, and 50 multicast receivers) and plot the various performance metrics in a graph. We are interested in per-receiver throughput, packet latency, and overall network load. The per-receiver throughput is the average number of data bytes received per receiver, and ideally scales linearly with the number of multicast senders. The *Max* curve shows the ideal per-receiver throughput, assuming a 100% packet delivery ratio.

### 7.3.1. BCAST under medium load




| | 1 | 2 | 5 | 10 |
|---|---|---|---|---|
| 02 Mbps | 1769601.27 | 3523629.957 | 8398183.219 | 8652084.019 |
| 11 Mbps | 1775153.234 | 3548610.232 | 8823916.954 | 17552901.73 |
| 36 Mbps | 1776040.55 | 3551047.68 | 8846580.941 | 17662515.61 |
| Max | 1781760 | 3563520 | 8908800 | 17817600 |
| | Number of Senders | | | |

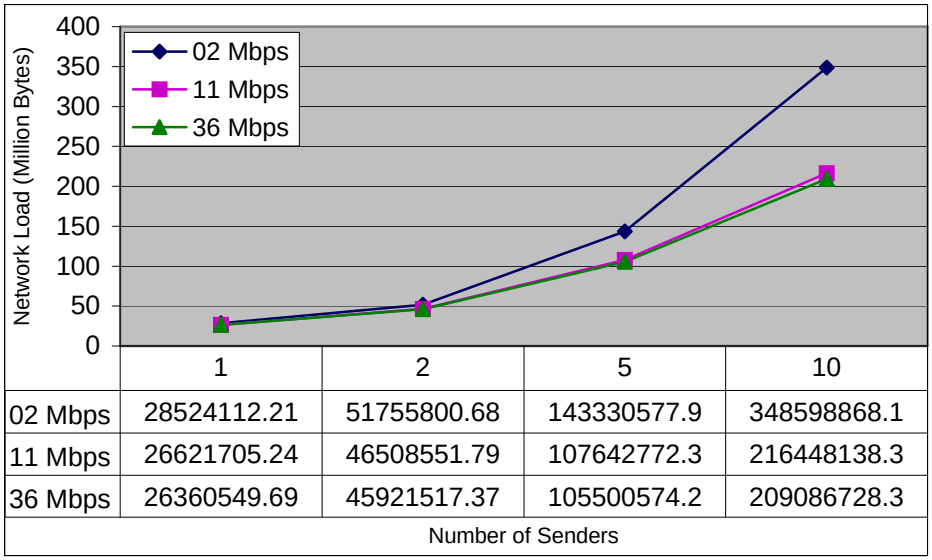
***Figure 1: BCAST throughput, 01 m/s maximum speed, medium load***




| | 1 | 2 | 5 | 10 |
|---|---|---|---|---|
| 02 Mbps | 1766511.698 | 3518990.254 | 8421666.816 | 8566523.904 |
| 11 Mbps | 1769344.696 | 3537499.177 | 8812299.878 | 17536901.53 |
| 36 Mbps | 1769501.491 | 3538553.979 | 8833698.816 | 17625989.53 |
| Max | 1781760 | 3563520 | 8908800 | 17817600 |
| | Number of Senders | | | |

***Figure 2: BCAST throughput, 20 m/s maximum speed, medium load***

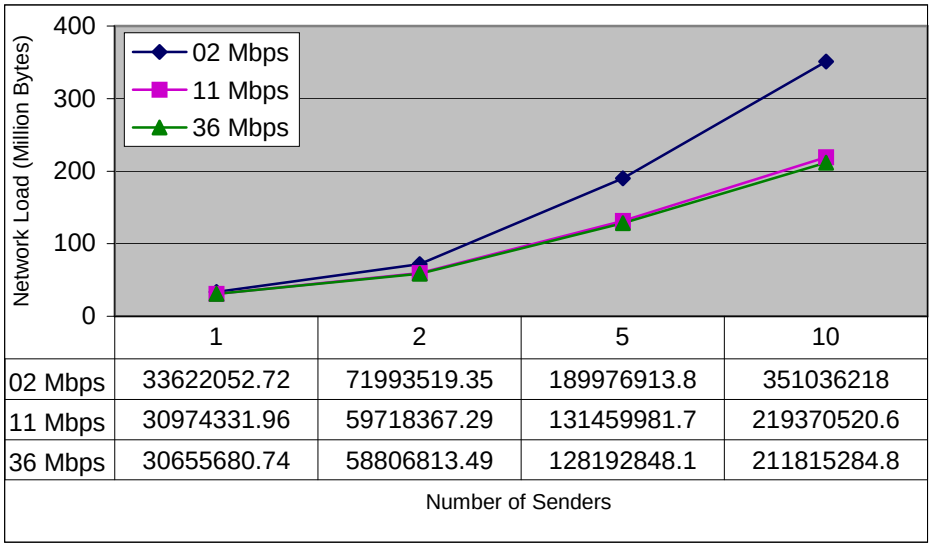
Figures 1 and 2 show that, as expected, with an 11 Mbps and 36 Mbps MAC layer, BCAST achieves close to maximal throughput, since the average packet delivery ratio is consistently high. For a 2 Mbps MAC layer, BCAST's throughput drops slightly for the 5 senders scenarios, and remains at about that level for the 10 senders scenarios (the packet delivery ratios drop to below 50% for the 10 sender scenarios). The results are very similar for low and high mobility, with a slightly better performance of the protocol under low mobility.




| | 1 | 2 | 5 | 10 |
|---|---|---|---|---|
| 02 Mbps | 28524112.21 | 51755800.68 | 143330577.9 | 348598868.1 |
| 11 Mbps | 26621705.24 | 46508551.79 | 107642772.3 | 216448138.3 |
| 36 Mbps | 26360549.69 | 45921517.37 | 105500574.2 | 209086728.3 |

Number of Senders

***Figure 3: BCAST network load, 01 m/s maximum speed, medium load***




| | 1 | 2 | 5 | 10 |
|---|---|---|---|---|
| 02 Mbps | 33622052.72 | 71993519.35 | 189976913.8 | 351036218 |
| 11 Mbps | 30974331.96 | 59718367.29 | 131459981.7 | 219370520.6 |
| 36 Mbps | 30655680.74 | 58806813.49 | 128192848.1 | 211815284.8 |

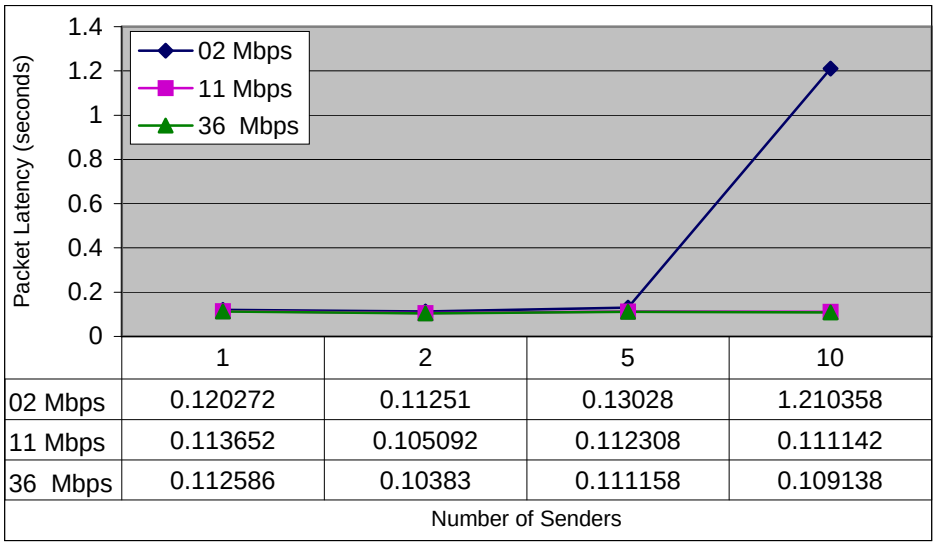
Number of Senders

***Figure 4: BCAST network load, 20 m/s maximum speed, medium load***

Figures 3 and 4 plot the network load (the total number of bytes transmitted at the MAC layer) for BCAST with different MAC layers and mobility rates. Even though a 2 Mbps MAC layer results in lower throughput, the network load is higher. The load is lowest for the 36 Mbps MAC layer. As all packet transmissions (the periodic HELLO messages as well as the data packets) are broadcast at the MAC layer, MAC-layer retransmission attempts are *not* the cause of the increased traffic load: as MAC-layer broadcasts are unacknowledged, a sender is unable to determine packet loss and retry, unlike unicast transmissions. Rather, with a lower-bandwidth MAC layer, more packet transmissions collide with other packets and are lost. Loosing a data packet to collisions has a number of effects: first, the multicast receivers may not receive the packet. As there are multiple paths over which a data packet can be transmitted from multicast sender to the multicast receivers, the protocol is robust enough to recover from a small number of such collisions losses. However, it does require additional data packet broadcasts by other nodes.

Second, while a node buffers a data packet for retransmission, the protocol listens to other transmissions in its neighborhood. If it determines that another broadcast will cover all its uncovered neighbors, it will cancel its pending retransmission. As more data packets get lost due to collisions, nodes will be provided with fewer optimization opportunities (i.e., they may have to broadcast a data packet to cover all uncovered neighbors), again increasing the network load. In the limiting case, BCAST will behave like FLOOD: each node will have to rebroadcast a data packet. From previous experiments we know that the network load of FLOOD is up to twice as high as BCAST's network load, with no increased packet delivery performance.

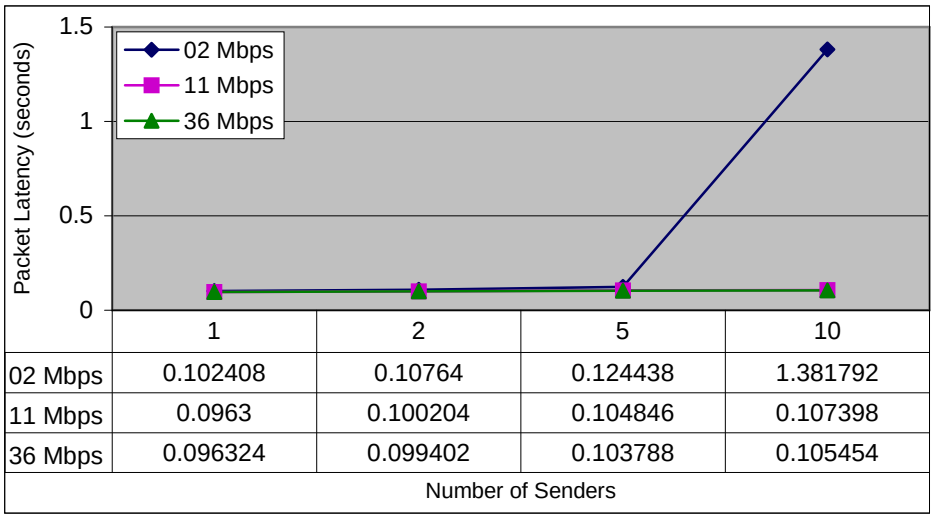
Figures 5 and 6 plot the average packet latency. Overall, per-packet latency is relatively constant, slightly above 100 ms. Only in scenarios with 10 senders over a low-bandwidth MAC layer does packet latency increase significantly, to 1.2 to 1.4 seconds. A closer look at the simulation results also reveals that packet latency is a bit shorter under high mobility and faster MAC layer, but the differences are relatively minor. Overall, the prime determinant of packet latency is the random per-node delay before a data packet is rebroadcast. In our implementation of BCAST, packet transmission is delayed by an amount of time randomly chosen from [0, MAX/N * 100 ms], where MAX is the maximum neighbor degree of any neighbor node and N is the node's own degree.




| | 1 | 2 | 5 | 10 |
|---|---|---|---|---|
| 02 Mbps | 0.120272 | 0.11251 | 0.13028 | 1.210358 |
| 11 Mbps | 0.113652 | 0.105092 | 0.112308 | 0.111142 |
| 36 Mbps | 0.112586 | 0.10383 | 0.111158 | 0.109138 |

Number of Senders

***Figure 5: BCAST packet latency, 01 m/s maximum speed, medium load***



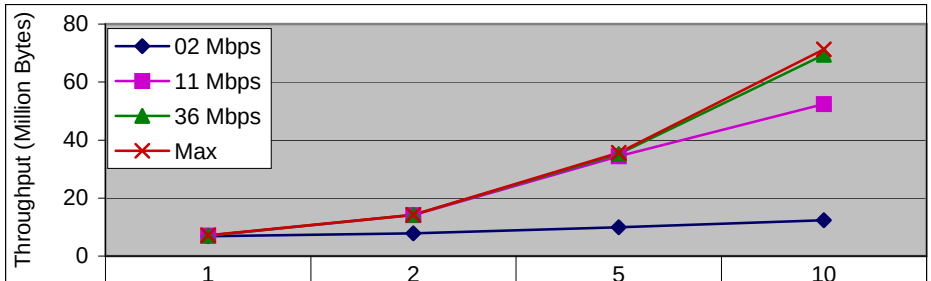

| | 1 | 2 | 5 | 10 |
|---|---|---|---|---|
| 02 Mbps | 0.102408 | 0.10764 | 0.124438 | 1.381792 |
| 11 Mbps | 0.0963 | 0.100204 | 0.104846 | 0.107398 |
| 36 Mbps | 0.096324 | 0.099402 | 0.103788 | 0.105454 |

Number of Senders

***Figure 6: BCAST packet latency, 20 m/s maximum speed, medium load***

### 7.3.2. BCAST under heavy load

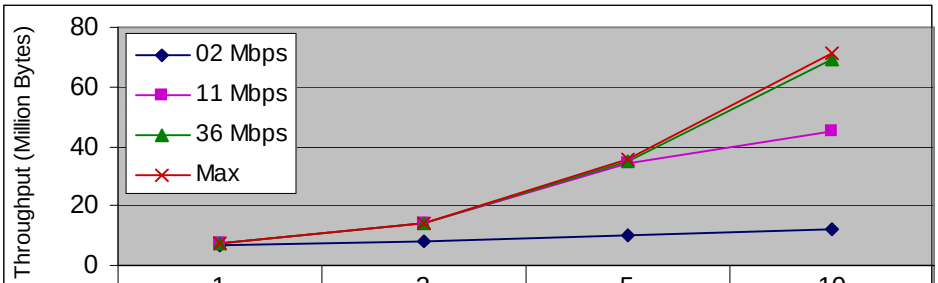
Under heavy load, twice as many data packets, each twice the size, compared to medium load, are transmitted by each multicast sender. Figures 7 and 8 plot the per-receiver throughput, indicating that low-bandwidth MAC layers reduce the protocol performance significantly as the number of multicast sources increases. Only the 36 Mbps MAC layer results in consistently high packet delivery ratios (and therefore per-node throughputs). Mobility seems to have a slightly negative effect on protocol performance.




| | 1 | 2 | 5 | 10 |
|---|---|---|---|---|
| 02 Mbps | 6828659.343 | 7796839.219 | 9986123.366 | 12361423.26 |
| 11 Mbps | 7081455.452 | 14125878.8 | 34404859.08 | 52422515.1 |
| 36 Mbps | 7096864.113 | 14179987.29 | 35146641.41 | 69369903.51 |
| Max | 7127040 | 14254080 | 35635200 | 71270400 |

Number of Senders

***Figure 7: BCAST throughput, 01 m/s maximum speed, heavy load***

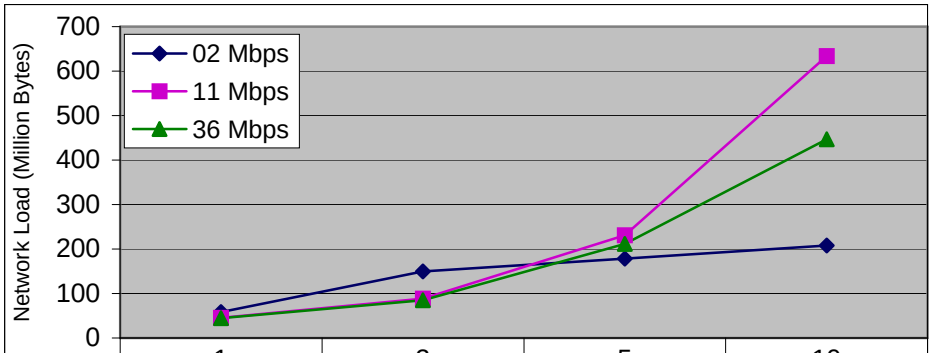



| | 1 | 2 | 5 | 10 |
|---|---|---|---|---|
| 02 Mbps | 6883922.412 | 7879455.867 | 10283392.2 | 12342180.25 |
| 11 Mbps | 7062141.174 | 14079923.65 | 34397660.77 | 44732581.48 |
| 36 Mbps | 7067743.027 | 14125850.3 | 35109794.61 | 69370616.22 |
| Max | 7127040 | 14254080 | 35635200 | 71270400 |

Number of Senders

***Figure 8: BCAST throughput, 20 m/s maximum speed, heavy load***

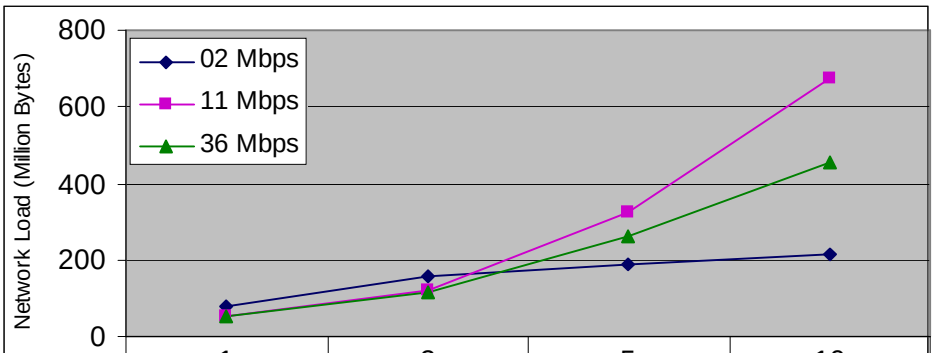
Figures 9 and 10 show the network load under heavy data traffic. As discussed before, a higher per-node throughput does not necessarily come at the cost of a higher network load. For the low-bandwidth 2 Mbps MAC layer, total network load seems to hit a limit at about 210 Million Bytes. As we ran the simulations for 900 seconds, with user data being generated and transmitted after the initial 30 seconds, this translates into a network load rate of approximately 1.9 Mbps, close to the nominal link rate. This number is overly optimistic, since we do not account for packet collisions, but the rough estimate shows that the network capacity limit is reached. For the higher-bandwidth MAC layers, we use a much smaller fraction of the nominal link rate. However, for the 11 Mbps MAC layer, a significant number of packet collisions occurs, resulting in lost opportunities to optimize the packet re-transmissions. Only the 36 Mbps MAC layer provides enough capacity to allow the protocol optimizations to work, resulting in comparatively low network load and high packet delivery ratio. For example, in the 50 receivers scenarios, each packet transmission at the MAC layer (data packets or HELLO packets) results in the reception of about 4 data packets at the user level.




| | 1 | 2 | 5 | 10 |
|---|---|---|---|---|
| 02 Mbps | 58202104.72 | 149497581.3 | 178774788.9 | 207856283.2 |
| 11 Mbps | 45329296.45 | 88178371.78 | 230769026.3 | 633884129.9 |
| 36 Mbps | 44327185.17 | 85118951.12 | 211279222.3 | 446567229.7 |

Number of Senders

***Figure 9: BCAST network load, 01 m/s maximum speed, heavy load***

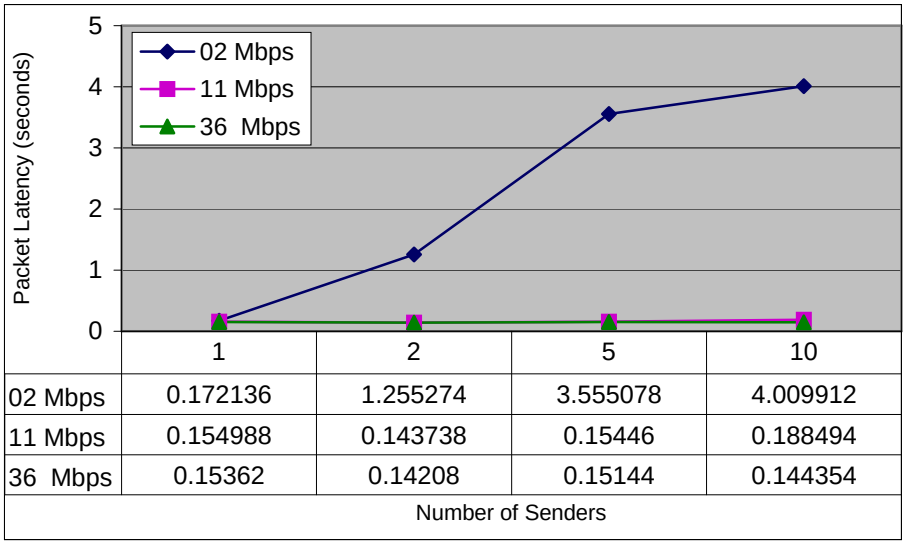



| | 1 | 2 | 5 | 10 |
|---|---|---|---|---|
| 02 Mbps | 79214627.61 | 156370921.4 | 186032600.5 | 212114966.5 |
| 11 Mbps | 54633831.13 | 119480607.5 | 326112342.3 | 676255247.8 |
| 36 Mbps | 53139757.1 | 112458504 | 263252421.2 | 453657155.4 |

Number of Senders

***Figure 10: BCAST network load, 20 m/s maximum speed, heavy load***

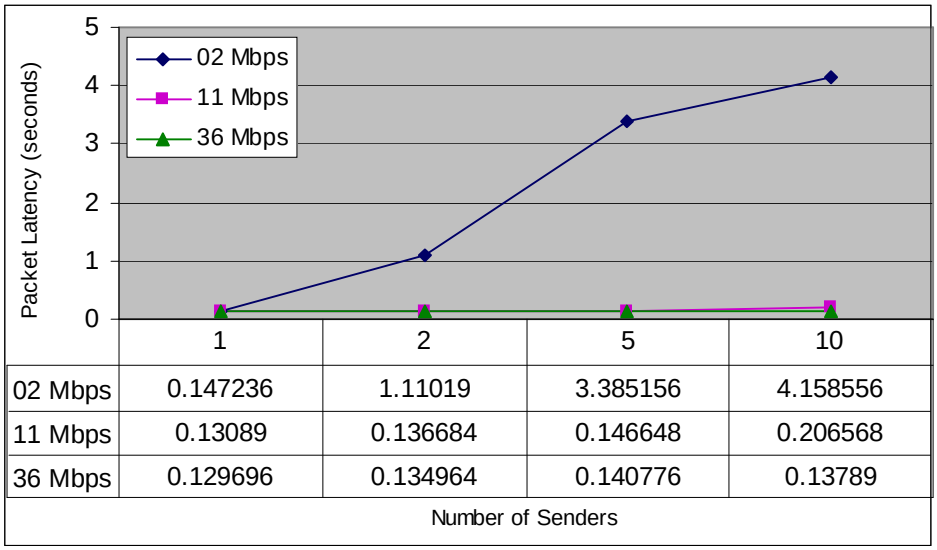
Finally, Figures 11 and 12 summarize the packet latency. As expected, for the low-bandwidth MAC layer, packet latency grows to multiple seconds as the number of multicast senders increases. As the network is operating at its capacity, packets are queued at the MAC layer for a long time before a node has the chance to transmit them, dwarfing the protocol-induced packet delays. For higher-bandwidth MAC layers, these queuing effects are less pronounced but still visible when comparing the packet latencies to the results under medium load (Figures 5 and 6). Compared to those numbers, packet latencies are 35% to 100% higher for the heavy load scenarios.




| | 1 | 2 | 5 | 10 |
|---|---|---|---|---|
| 02 Mbps | 0.172136 | 1.255274 | 3.555078 | 4.009912 |
| 11 Mbps | 0.154988 | 0.143738 | 0.15446 | 0.188494 |
| 36 Mbps | 0.15362 | 0.14208 | 0.15144 | 0.144354 |

Number of Senders

***Figure 11: BCAST packet latency, 01 m/s maximum speed, heavy load***




| | 1 | 2 | 5 | 10 |
|---|---|---|---|---|
| 02 Mbps | 0.147236 | 1.11019 | 3.385156 | 4.158556 |
| 11 Mbps | 0.13089 | 0.136684 | 0.146648 | 0.206568 |
| 36 Mbps | 0.129696 | 0.134964 | 0.140776 | 0.13789 |

Number of Senders

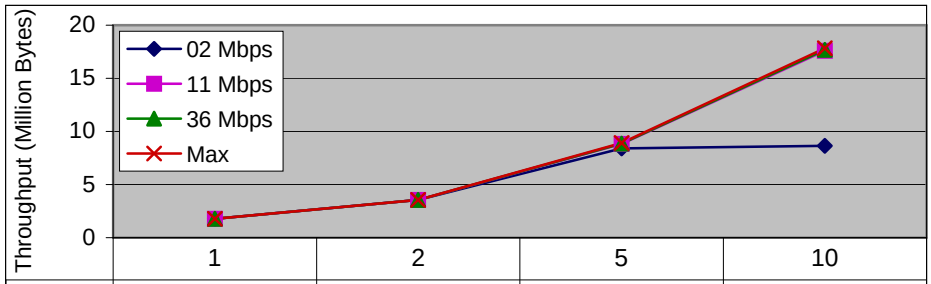
***Figure 12: BCAST packet latency, 20 m/s maximum speed, heavy load***

In summary, the protocol performs as expected. With more capacity at the MAC layer (i.e., higher data rate), the per-received throughput is higher and packet latency is lower. This is true for both medium and high traffic loads. We also noticed that as the limits of the MAC layer are approached, the protocol performance begins to suffer due to increased MAC layer packet collisions. These collisions potentially reduce the packet delivery ratio and in particular reduce the chances of the protocol to optimize the packet re-transmission by overhearing other broadcasts. So more nodes will re-transmit data packets, resulting in increased network traffic, further reducing protocol performance. The protocol is relatively stable with respect to mobility and observed performance differences are probably more closely related to exceeding the network capacity under different scenarios. Focusing on the 36 Mbps MAC layer only, it seems that BCAST has slightly higher packet delivery ratios under low mobility, though this comes at the cost of a slightly increased packet latency.

## 7.4. Reliable BCAST performance

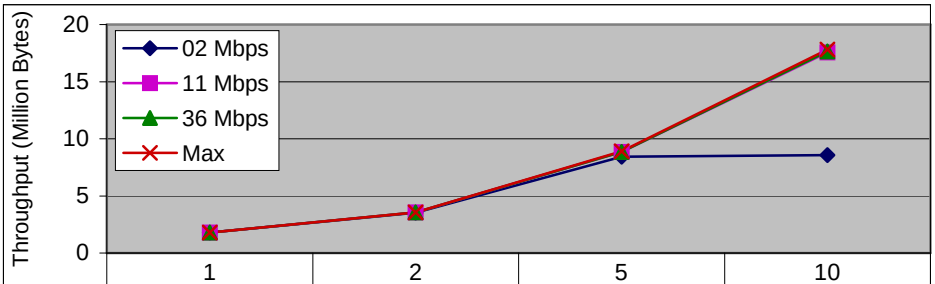
Similar to the best-effort BCAST protocol, we ran the reliable BCAST protocol under medium and heavy load and various MAC layers. Compared to the best-effort protocol, the reliable protocol uses a NACK-based packet retransmission scheme: all packets transmitted by a multicast sender have a unique sequence number. As nodes receive data packets, they buffer them in a FIFO buffer of limited size (10 packets). When a node receives packet N from sender S, but has not yet received packet N-1 from that same sender, it will broadcast a negative acknowledgement (NACK) to its immediate neighbors, requesting retransmission of that packet. All neighbors with packet N-1 from sender S in their cache will schedule a retransmission of that packet with random delay. If a node with pending retransmission overhears another retransmission of that packet, it will cancel its transmission attempt, otherwise it will broadcast the packet once its timer expires. In a heavily congested network, these NACKs and packet retransmissions will negatively impact overall protocol performance, so NACKs are throttled (i.e., when a node observes a high amount of network traffic in the recent past, it will suppress NACKs). Ideally, the appropriate parameter settings for this throttling mechanism should be derived at experimentally for each MAC layer. However, for the time being we use the parameters derived for the 02 Mbps MAC: a node considers the network busy and therefore suppresses NACKs when it overheard more than 30 packet transmissions in the last 400 milliseconds (including its own transmissions). The following subsections summarize and analyze our findings.

### 7.4.1. Reliable BCAST under Medium Load




| | 1 | 2 | 5 | 10 |
|---|---|---|---|---|
| 02 Mbps | 1776685.548 | 3548938.076 | 8411778.048 | 8653723.238 |
| 11 Mbps | 1776942.121 | 3553506.509 | 8831489.434 | 17580233.93 |
| 36 Mbps | 1776963.502 | 3553613.414 | 8852175.667 | 17688493.67 |
| Max | 1781760 | 3563520 | 8908800 | 17817600 |
| Number of Senders | | | | |

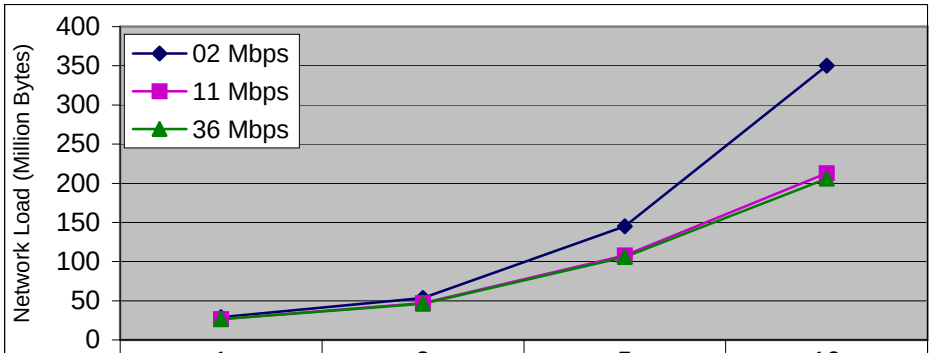
***Figure 13: Reliable BCAST throughput, 01 m/s maximum speed, medium load***




| | 1 | 2 | 5 | 10 |
|---|---|---|---|---|
| 02 Mbps | 1779906.97 | 3533144.556 | 8434531.123 | 8569552.896 |
| 11 Mbps | 1780473.569 | 3558694.994 | 8843195.597 | 17553650.07 |
| 36 Mbps | 1780452.188 | 3559186.76 | 8861993.165 | 17650791.63 |
| Max | 1781760 | 3563520 | 8908800 | 17817600 |
| Number of Senders | | | | |

***Figure 14: Reliable BCAST throughput, 20 m/s maximum speed, medium load***

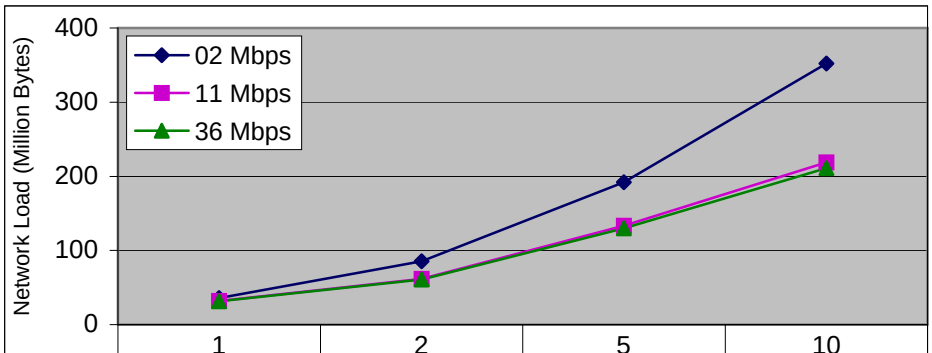
Figures 13 and 14 plot the per-receiver throughput of the reliable BCAST protocol over various MAC layers under medium load. Similar to the best-effort protocol version, the per-sender throughput is close to the maximum throughput for an 11 Mbps or a 36 Mbps MAC layer. Only a 2 Mbps MAC layer causes a drop in throughput for 5 and in particular 10 senders, when the network is congested. The reliable protocol increases per-receiver throughput for all scenarios, under both low and high mobility, compared to the best-effort protocol. The improvement is higher for low-bandwidth MACs and for fewer multicast senders. For low-bandwidth MACs, the increased network load results in more packet losses due to MAC-layer collisions, providing more opportunities for the NACK scheme to increase packet delivery. However, for an increasing number of senders, and the ensuing increase in network load, this NACK mechanism is throttled, reducing its effectiveness.




| | 1 | 2 | 5 | 10 |
|---|---|---|---|---|
| 02 Mbps | 29040966.95 | 53632636.71 | 144876554 | 350132878.1 |
| 11 Mbps | 26687025.45 | 46795600.52 | 108028712.6 | 213013963.3 |
| 36 Mbps | 26418985.55 | 46071322.99 | 105716879 | 205722286.1 |

Number of Senders

***Figure 15: Reliable BCAST network load, 01 m/s maximum speed, medium load***

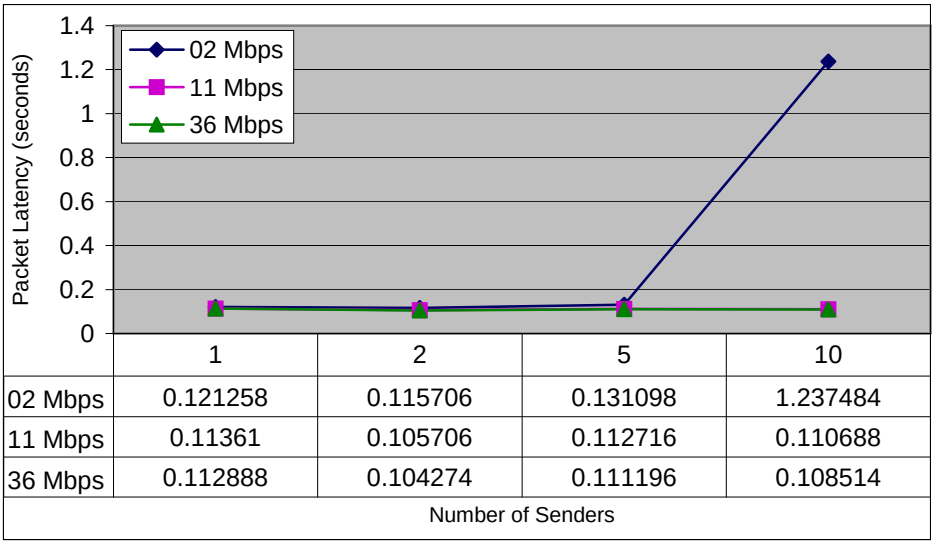



| | 1 | 2 | 5 | 10 |
|---|---|---|---|---|
| 02 Mbps | 35664434.36 | 85518135.79 | 191825612.4 | 351991642.8 |
| 11 Mbps | 32115879.09 | 61716634.53 | 133501103 | 218598906.3 |
| 36 Mbps | 31706081.57 | 60637747.68 | 130072807.2 | 210976665.6 |

Number of Senders

***Figure 16: Reliable BCAST network load, 20 m/s maximum speed, medium load***

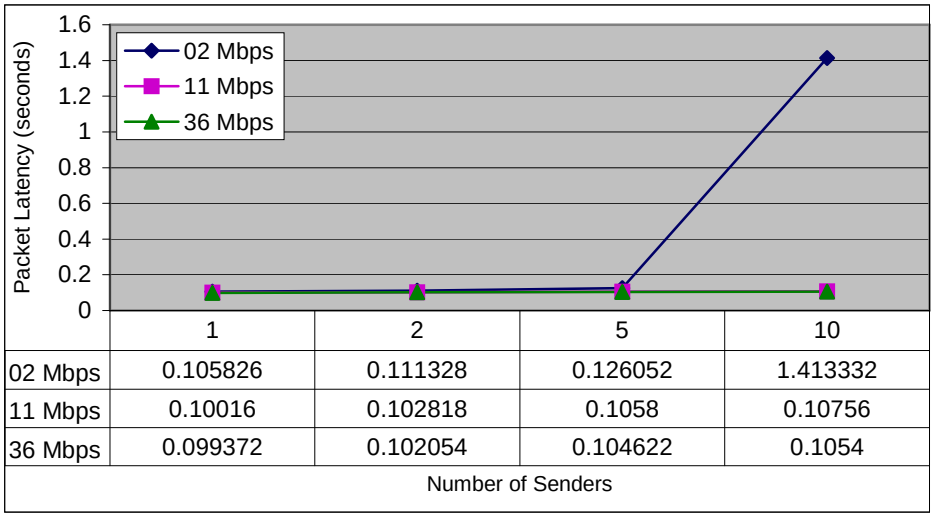
The network load plots (Figures 15 and 16) show trends similar to the best-effort protocol version: for lower-bandwidth MAC protocols, more MAC layer collisions result in a higher number of dropped packets, reducing the optimization potential of the protocol and resulting in an increased number of data packet re-transmissions. This in turn increases the network load and potentially leads to a further decrease in the packet delivery ratio. Since NACKs are throttled to prevent flooding the network, the retransmission mechanism in the reliable protocol cannot recover from these effects.




| | 1 | 2 | 5 | 10 |
|---|---|---|---|---|
| 02 Mbps | 0.121258 | 0.115706 | 0.131098 | 1.237484 |
| 11 Mbps | 0.11361 | 0.105706 | 0.112716 | 0.110688 |
| 36 Mbps | 0.112888 | 0.104274 | 0.111196 | 0.108514 |

Number of Senders

***Figure 17: Reliable BCAST packet latency, 01 m/s maximum speed, medium load***




| | 1 | 2 | 5 | 10 |
|---|---|---|---|---|
| 02 Mbps | 0.105826 | 0.111328 | 0.126052 | 1.413332 |
| 11 Mbps | 0.10016 | 0.102818 | 0.1058 | 0.10756 |
| 36 Mbps | 0.099372 | 0.102054 | 0.104622 | 0.1054 |

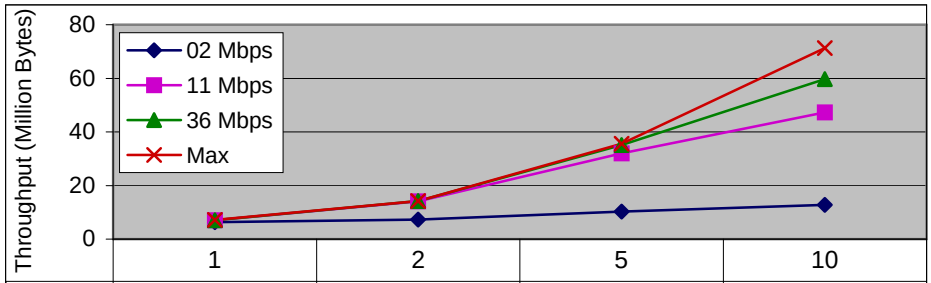
Number of Senders

***Figure 18: Reliable BCAST packet latency, 20 m/s maximum speed, medium load***

Figures 17 and 18 plot the average per-packet latency. Packet latency is slightly above 100 ms for almost all scenarios. As explained above, this is mostly determined by the random packet retransmission delay implemented in the protocol. For 10 multicast senders and a low-bandwidth MAC, with serious network congestion, packet latency increases to 1.2 to 1.4 seconds. Similar to the best-effort protocol, packet latencies are slightly lower for higher mobility scenarios and higher-bandwidth MAC layers. Also, the packet latencies for the reliable protocol are slightly higher compared to the best-effort version, since packet re-retransmissions, while increasing the packet delivery ratio and therefore per-receiver throughput, increase average packet latency.

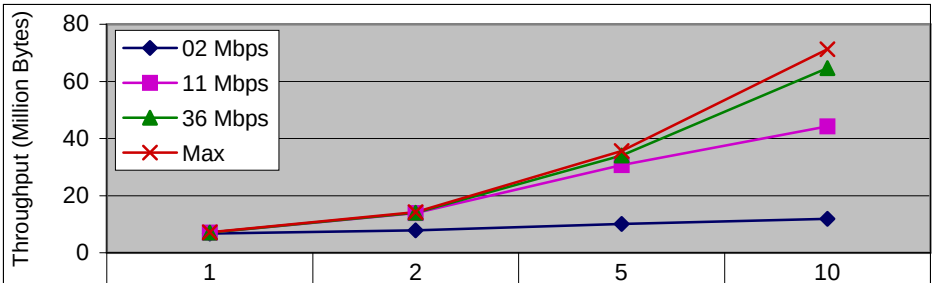
### 7.4.2. Reliable BCAST under heavy load

We also evaluated the reliable BCAST protocol under heavy traffic load, where each multicast sender transmits 8 packets per second, each of size 1024 bytes.




| | 1 | 2 | 5 | 10 |
|---|---|---|---|---|
| 02 Mbps | 6333686.415 | 7295922.34 | 10216184.22 | 12789900.9 |
| 11 Mbps | 7087470.674 | 14123199.04 | 32025354.24 | 47244435.46 |
| 36 Mbps | 7100498.903 | 14181327.18 | 35157759.59 | 59730439.37 |
| Max | 7127040 | 14254080 | 35635200 | 71270400 |
| Number of Senders | | | | |

***Figure 19: Reliable BCAST throughput, 01 m/s maximum speed, heavy load***

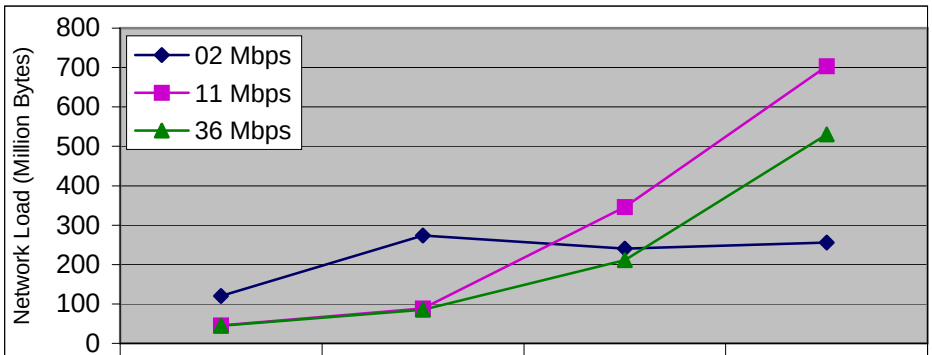



| | 1 | 2 | 5 | 10 |
|---|---|---|---|---|
| 02 Mbps | 6770573.967 | 7800716.329 | 10133724.36 | 11861247.59 |
| 11 Mbps | 7066887.782 | 13939435.44 | 30749542.81 | 44208316.42 |
| 36 Mbps | 7063694.868 | 14015951.34 | 34014012.21 | 64612461.77 |
| Max | 7127040 | 14254080 | 35635200 | 71270400 |
| Number of Senders | | | | |

***Figure 20: Reliable BCAST throughput, 20 m/s maximum speed, heavy load***

Figures 19 and 20 plot the per-receiver throughput with the various MAC layers, as well as the maximum throughput, assuming a 100% packet delivery ratio. The per-receiver throughput increases with the capacity of the MAC layer, as is expected. However, it is noticeable that the maximum throughput is lower for the reliable protocol than the best-effort protocol for the 10 multicast sender scenarios, see Figures 7 and 8. As discussed earlier, in heavily loaded networks, issuing too many NACKs, which in turn trigger additional packet retransmissions, may have a negative impact on the overall protocol performance. Therefore, the reliable protocol implements a NACK throttling mechanism. It seems from the results shown here that simply applying the parameters derived experimentally for the 2 Mbps MAC layer to other MAC layer data rates is not appropriate, reducing the protocol performance. Ideally, the throttling mechanism should

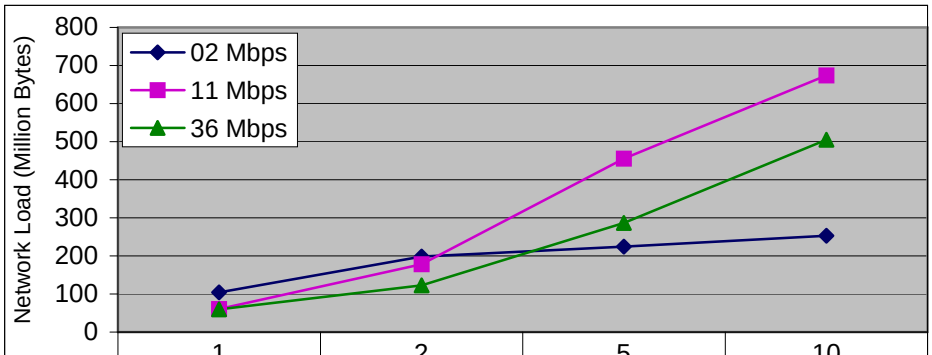
allow the reliable protocol version to perform nearly as well as the unreliable version in the worst case and result in improved packet delivery ratios in the majority of scenarios.




| | 1 | 2 | 5 | 10 |
|---|---|---|---|---|
| 02 Mbps | 119958448 | 273673458.7 | 240567334.7 | 255536810 |
| 11 Mbps | 45646610.1 | 88506201.52 | 346393223.9 | 703074454.5 |
| 36 Mbps | 44560066.09 | 85308929.9 | 211203211.9 | 530713119.3 |

Number of Senders

***Figure 21: Reliable BCAST network load, 01 m/s maximum speed, heavy load***



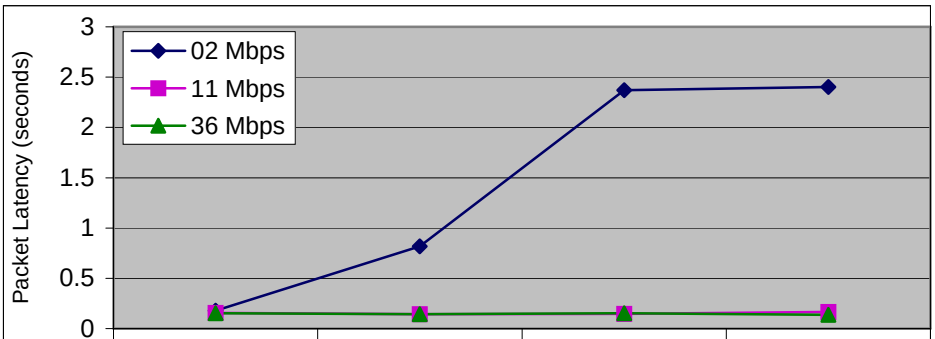

| | 1 | 2 | 5 | 10 |
|---|---|---|---|---|
| 02 Mbps | 104378130.6 | 198599927.9 | 224260462.3 | 252942434.3 |
| 11 Mbps | 60090423.49 | 178020921.7 | 455267355.7 | 673930138 |
| 36 Mbps | 60052375.44 | 123076529.5 | 286284497.2 | 505441790.8 |

Number of Senders

***Figure 22: Reliable BCAST network load, 20 m/s maximum speed, heavy load***

The total network load, measured in number of bytes transmitted at the MAC layer for all packets (data and control), is shown in Figures 21 and 22. Similar to the best-effort protocol, the network hits a hard capacity limit for the low-bandwidth MAC layer, which ultimately causes the poor protocol performance (i.e., many packets are dropped due to buffer overflows). For higher-bandwidth MAC layers, the number of collisions determines the total traffic load. For the 11 Mbps MAC layer, this number is higher, resulting in more data packet re-retransmissions, either directly because data packets are

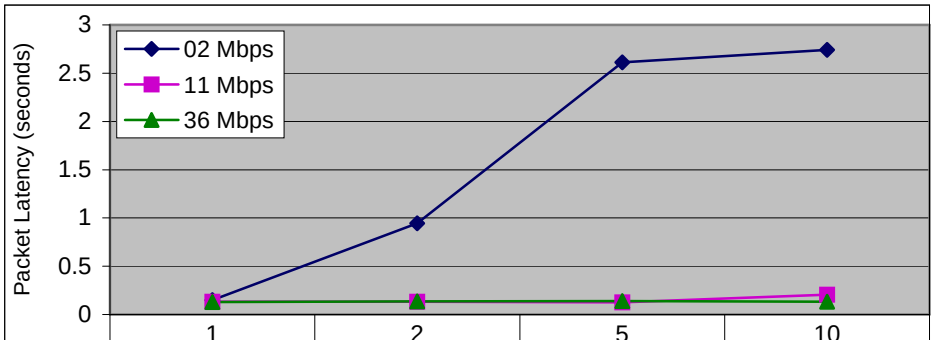
lost and therefore other nodes in the neighborhood retransmit the data packet or indirectly because nodes fail to learn about successful broadcasts and therefore cancel their potentially unnecessary retransmissions. Compared to the best-effort protocol version (see Figures 9 and 10), the total traffic load is higher, due to the NACK broadcasts and packet re-transmissions. Again, as the increased traffic load does not result in an increased per-node throughput or packet delivery ratio, this indicates that the NACK throttle is too weak for higher-bandwidth MACs.




| | 1 | 2 | 5 | 10 |
|---|---|---|---|---|
| 02 Mbps | 0.178826 | 0.81794 | 2.37167 | 2.402816 |
| 11 Mbps | 0.15515 | 0.143802 | 0.14709 | 0.16629 |
| 36 Mbps | 0.153806 | 0.142 | 0.151318 | 0.13765 |

Number of Senders

***Figure 23: Reliable BCAST packet latency, 01 m/s maximum speed, heavy load***




| | 1 | 2 | 5 | 10 |
|---|---|---|---|---|
| 02 Mbps | 0.15072 | 0.945638 | 2.612634 | 2.740616 |
| 11 Mbps | 0.131874 | 0.134012 | 0.127276 | 0.204064 |
| 36 Mbps | 0.130598 | 0.135762 | 0.137656 | 0.13368 |

Number of Senders

***Figure 24: Reliable BCAST packet latency, 20 m/s maximum speed, heavy load***

Figures 23 and 24 show the packet latency of reliable BCAST under heavy load. The dominant contributor to the packet latency is not the protocol-induced re-broadcast delay but queuing delays in accessing a busy MAC, at least for the low-bandwidth MAC. For higher-bandwidth MACs, the protocol-dependent delay is still the dominant component, but the MAC access delay starts to show as well. Comparing these figures to Figures 17

and 18, that show the packet latency under the same scenarios for medium load, the delay has increased by between 35% to up to 100%.

## 7.5. Reliable vs. best-effort BCAST

All else being equal, the packet delivery ratio of the best-effort BCAST protocol can be improved by adding a re-transmission component to the protocol (which results in the reliable BCAST protocol) or by providing more network capacity in the form of a higher-bandwidth MAC layer. This section looks at a comparison of the resulting performance.

***Table 20: PDR and latency for BCAST, 01 m/s max speed, 36 Mbps data rate, medium load***

| | 1 Sender | | 2 Senders | | 5 Senders | | 10 Senders | |
|---|---|---|---|---|---|---|---|---|
| | PDR | Latency | PDR | Latency | PDR | Latency | PDR | Latency |
| **10 Receivers** | 0.998 | 0.109 | 0.998 | 0.101 | 0.994 | 0.109 | 0.992 | 0.107 |
| **20 Receivers** | 0.998 | 0.113 | 0.998 | 0.104 | 0.994 | 0.112 | 0.992 | 0.110 |
| **30 Receivers** | 0.996 | 0.114 | 0.996 | 0.105 | 0.992 | 0.112 | 0.991 | 0.110 |
| **40 Receivers** | 0.996 | 0.115 | 0.996 | 0.106 | 0.992 | 0.112 | 0.991 | 0.110 |
| **50 Receivers** | 0.996 | 0.112 | 0.996 | 0.103 | 0.992 | 0.110 | 0.991 | 0.108 |

***Table 21: PDR and latency for BCAST, 20 m/s max speed. 36 Mbps data rate, medium load***

| | 1 Sender | | 2 Senders | | 5 Senders | | 10 Senders | |
|---|---|---|---|---|---|---|---|---|
| | PDR | Latency | PDR | Latency | PDR | Latency | PDR | Latency |
| **10 Receivers** | 0.993 | 0.098 | 0.993 | 0.100 | 0.992 | 0.104 | 0.990 | 0.105 |
| **20 Receivers** | 0.993 | 0.097 | 0.993 | 0.100 | 0.991 | 0.104 | 0.989 | 0.106 |
| **30 Receivers** | 0.993 | 0.097 | 0.993 | 0.100 | 0.992 | 0.104 | 0.989 | 0.106 |
| **40 Receivers** | 0.993 | 0.096 | 0.993 | 0.100 | 0.992 | 0.104 | 0.989 | 0.106 |
| **50 Receivers** | 0.993 | 0.094 | 0.993 | 0.098 | 0.992 | 0.102 | 0.990 | 0.104 |

Overall, a higher-rate MAC layer results in very good performance for best-effort BCAST. Tables 20 and 21 summarize PDR and packet latency for best-effort BCAST over a 36 Mbps MAC layer and medium traffic load. Tables 22 and 23 show the results of reliable BCAST over a 2 Mbps MAC under light load. The performance of the reliable BCAST protocol is inferior for the 10 sender scenarios, where the network experiences severe congestion and the NACK-based reliability mechanism is throttled to prevent control packets from flooding the network. Under low mobility and for relatively fewer senders, the packet delivery ratios are roughly similar, and packet latency is slightly better for the best-effort protocol. Only under high mobility and for relatively few senders does the extra effort to increase the packet delivery ratio (buffering packets, NACK-based scheme, etc.) result in a slightly increased packet delivery ratio.

***Table 22: PDR and latency for reliable BCAST, 01 m/s max speed, 2 Mbps data rate, light load***

| | 1 Sender | | 2 Sender | | 5 Sender | | 10 Sender | |
|---|---|---|---|---|---|---|---|---|
| | PDR | Latency | PDR | Latency | PDR | Latency | PDR | Latency |
| **10 Receivers** | 0.998 | 0.117 | 0.998 | 0.109 | 0.995 | 0.126 | 0.972 | 0.130 |
| **20 Receivers** | 0.998 | 0.120 | 0.998 | 0.113 | 0.995 | 0.129 | 0.972 | 0.134 |
| **30 Receivers** | 0.997 | 0.122 | 0.997 | 0.114 | 0.993 | 0.129 | 0.970 | 0.134 |
| **40 Receivers** | 0.996 | 0.123 | 0.996 | 0.114 | 0.992 | 0.129 | 0.970 | 0.134 |
| **50 Receivers** | 0.997 | 0.119 | 0.997 | 0.111 | 0.993 | 0.126 | 0.970 | 0.131 |

*Table 23: PDR and latency for reliable BCAST, 20 m/s max speed, 2 Mbps data rate, light load*

| | 1 Sender | | 2 Sender | | 5 Sender | | 10 Sender | |
|---|---|---|---|---|---|---|---|---|
| | PDR | Latency | PDR | Latency | PDR | Latency | PDR | Latency |
| **10 Receivers** | 0.999 | 0.109 | 0.999 | 0.113 | 0.996 | 0.125 | 0.970 | 0.132 |
| **20 Receivers** | 0.999 | 0.108 | 0.999 | 0.113 | 0.995 | 0.126 | 0.970 | 0.132 |
| **30 Receivers** | 0.999 | 0.108 | 0.999 | 0.113 | 0.996 | 0.122 | 0.970 | 0.132 |
| **40 Receivers** | 0.999 | 0.107 | 0.999 | 0.112 | 0.996 | 0.122 | 0.970 | 0.132 |
| **50 Receivers** | 0.999 | 0.105 | 0.999 | 0.110 | 0.996 | 0.120 | 0.971 | 0.129 |

Tables 24 and 25 summarize the performance of the reliable protocol over a MAC layer with 36 Mbps data rate and medium traffic load. Compared to the performance of the reliable protocol under light load (Tables 22 and 23), the most obvious improvement occurs for the 10 senders scenarios, where the 36 Mbps MAC does not suffer from congestion. For scenarios with relatively fewer senders, the protocol performance (packet delivery ratio and per packet latency) is comparable. Compared to the best-effort version of the protocol (Tables 20 and 21), the packet delivery ratio improvement is very marginal under low mobility, and more noticeable under high mobility, in particular for scenarios with relatively few senders.

*Table 24: PDR and latency for rel. BCAST, 01 m/s max speed, 36 Mbps data rate, medium load*

| | 1 Sender | | 2 Senders | | 5 Senders | | 10 Senders | |
|---|---|---|---|---|---|---|---|---|
| | PDR | Latency | PDR | Latency | PDR | Latency | PDR | Latency |
| **10 Receivers** | 0.998 | 0.110 | 0.998 | 0.101 | 0.995 | 0.110 | 0.994 | 0.107 |
| **20 Receivers** | 0.998 | 0.113 | 0.998 | 0.105 | 0.995 | 0.112 | 0.994 | 0.110 |
| **30 Receivers** | 0.997 | 0.114 | 0.997 | 0.106 | 0.993 | 0.112 | 0.992 | 0.110 |
| **40 Receivers** | 0.996 | 0.115 | 0.996 | 0.106 | 0.993 | 0.112 | 0.992 | 0.110 |
| **50 Receivers** | 0.997 | 0.112 | 0.997 | 0.103 | 0.993 | 0.110 | 0.992 | 0.107 |

*Table 25: PDR and latency for rel. BCAST, 20 m/s max speed. 36 Mbps data rate, medium load*

| | 1 Sender | | 2 Senders | | 5 Senders | | 10 Senders | |
|---|---|---|---|---|---|---|---|---|
| | PDR | Latency | PDR | Latency | PDR | Latency | PDR | Latency |
| **10 Receivers** | 0.999 | 0.100 | 0.999 | 0.103 | 0.995 | 0.105 | 0.991 | 0.106 |
| **20 Receivers** | 0.999 | 0.100 | 0.999 | 0.102 | 0.995 | 0.105 | 0.990 | 0.106 |
| **30 Receivers** | 0.999 | 0.100 | 0.999 | 0.102 | 0.995 | 0.105 | 0.991 | 0.106 |
| **40 Receivers** | 0.999 | 0.099 | 0.999 | 0.102 | 0.995 | 0.105 | 0.991 | 0.106 |
| **50 Receivers** | 0.999 | 0.098 | 0.999 | 0.100 | 0.995 | 0.103 | 0.991 | 0.104 |

## 7.6. Conclusions

This section sheds some light on the influence of the MAC protocol on the performance of the best-effort and reliable versions of BCAST. We ran experiments with medium and heavy traffic loads over three distinct variations of the IEEE 802.11 MAC protocol family and summarized our results. The results are preliminary in that we have not fully optimized the protocol implementations: by prioritizing NACK, HELLO, and data packets, we may achieve better

performance for both protocols (prioritizing HELLO messages would enable nodes to have more accurate and timely neighborhood information, for example). Similarly, the NACK throttle mechanism should be tuned for different MAC protocols, the above results seem to indicate that applying the experimentally derived settings from the 2 Mbps MAC layer causes performance degradation over other MAC layers.

However, a number of results/trends are clearly visible, some straightforward:

- The results confirm that the MAC layer has a substantial impact on the protocol performance: both protocols perform worse under low-rate MACs. The impact of the MAC shows not only in performance metrics such as Packet Delivery Ratio and packet latency, but also in the network load created by the protocols. As the network becomes more congested, the protocol works less well (more packet retransmissions), adding to the network congestion, with detrimental effect on the performance metrics.

- Mobility is not a problem. Both protocols achieve fairly similar performance under both low and high mobility. High-mobility scenarios seem to result in slightly higher packet delivery ratios and lower packet latencies, in particular for medium traffic loads. Also, the reliable protocol seems to improve more on the unreliable protocol for comparable scenarios under high mobility and for relatively few multicast senders.

- The best-effort BCAST protocol achieves pretty good performance if collisions can be avoided. So additional ways to reduced collisions even for low-bandwidth MACs could be beneficial for the protocol. One idea would be to increase the random packet jitter in heavy network load scenarios, i.e., make the random jitter a function of observed network load (which we observe to throttle NACKs already). Alternatively, MAC protocols that provide medium access among neighboring nodes in a more coordinated fashion could improve the protocol performance. As shown in [Kanodia 2001], the number of collisions and the end-to-end delay can be significantly reduced even with partial coordination only, i.e., not all nodes in the neighborhood cooperate. Their results show that, compared to no coordination, if only 80% of nodes in a neighborhood participate in the coordination effort, the delay is reduced by 80% and the number of collisions decreases by over 50%. To achieve this, it may be possible to piggyback appropriate control information onto the periodic HELLO messages.

- Similarly, the reliability mechanism is really only advantageous in a limited number of scenarios. Comparing, for example, the performance of the best-effort version with the reliable version under medium traffic load and a 36 Mbps MAC layer (Tables 1, 2, 5, and 6), the performance improvements are rather incremental. When increasing the network traffic, the advantage of our NACK-based reliability mechanism decreases, as we have to throttle the transmission of NACKs to prevent overloading the network, with ensuing negative impact on the protocol performance.

As MAC rates increase for current and future networks, MANETs will be able to support a non-trivial amount of traffic per multicast sender. Achieving high packet delivery ratios in these networks could be done based on a retransmission scheme similar to the one we used. However, a more advantageous route appears to be to adjust the data volume through flow control to operate in the protocol "sweet spot", using the best -effort protocol as basic protocol. This could be achieved either by implementing a TCP-like flow control mechanism on top of the multicast routing protocol (as proposed by some researchers) or by providing information about the network performance to the multicast senders, which can adjust their data rate accordingly, for example by employing transcoding techniques to reduce the fidelity of a video or audio stream.

# 8. Summary

This report provides an in-depth study of one-to-many and many-to-many communication in mobile ad-hoc networks. The original goal of this work was to design an efficient protocol that delivers packets from one or multiple senders to many receptions with high probability. We started this effort by exploring the performance of a number of best-effort protocols: 2 unicast routing protocols, 3 multicast routing protocols, and 2 broadcast protocols. The extensive simulation results in Section 4 show that broadcast protocols perform surprisingly well, and that this performance does not come with a high overhead. So we decided to use the more efficient broadcast protocol, BCAST, as starting point for the next step. After exploring a number of design alternatives (Section 5), we enhanced BCAST with a NACK-based retransmission scheme to further increase the packet delivery ratio. The simulation results reported in Section 6 demonstrate that this mechanism indeed increases the packet delivery ratio. The experiments also show that a high degree of mobility is actually advantageous: as network partitions are potentially short-lived, our retransmission scheme is more likely to successfully recover from packet losses during such partitions. In contrast, in networks with longer-lived partitions, the amount of packets buffered at nodes needs to be increased substantially to recover in these cases. Finally, the results demonstrate that implementing any reliability mechanism has to be done with care. As the network capacity is limited, flooding the network with NACKs and the ensuing packet re-transmission attempts will have a detrimental impact on the protocol performance when the network is experiencing congestion. In the reliable BCAST protocol, we therefore have each node monitor the local network traffic and suppress NACKs when it observed too much traffic in the recent past.

In a last step, we explored the impact of the MAC layer on the performance of both best-effort BCAST and reliable BCAST. Varying the user traffic load and the MAC layer, Section 7 discusses a number of insights into the relationship between MAC and ROUTING layer. In particular, we noted that BCAST suffers as the number of MAC-level packet collisions increases. Therefore, approaches to reduce such collisions have the potential to increase the performance of BCAST substantially. The results also indicate that our NACK-based packet retransmission scheme increases packet delivery ratio in only a limited number of scenarios. If the user traffic, relative to the MAC layer data rate, is low, best-effort BCAST performs already extremely well. If, on the other hand, the network starts to experience congestion due to high user traffic, we have to throttle NACKs to prevent them from negatively impacting the protocol performance, so again there is little difference between the best-effort and reliable protocol versions. As the discussion in Section 7 also demonstrates, the NACK throttle has to be tuned carefully for the MAC layer, otherwise the resulting performance suffers.

In conclusion, BCAST is a protocol that achieves high packet delivery, at the cost of an increase in packet latency. We have shown that the protocol performs well in a wide range of scenarios and over a number of MAC layers (all of which were variants of the 802.11 protocol family). Increasing packet delivery through a retransmission scheme is, however, only of limited value. As MAC rates increase for current and future networks, MANETs will be able to support a non-trivial amount of traffic per multicast sender. Achieving high packet delivery ratios in these networks can be achieved by adjusting the data volume through flow control to operate in the protocol “sweet spot”, using the best -effort protocol as basic protocol.

# References


[Adamson 2002] B. Adamson, G. Borman, M. Handley, and J. Macker, *NACK-Oriented Reliable Multicast Protocol (NORM)*, IETF INTERNET-DRAFT draft-ietf-rmt-pi-norm-06.txt, expires September 2003 (work in progress).

[Anastasi 2001] G. Anastasi, A. Bartoli, and F. Spadoni, *A Reliable Multicast Protocol for Distributed Mobile Systems: Design and Evaluation*, IEEE Trans. on Parallel and Distributed Systems, pp. 1009-1022, Vol. 12, No. 10, Oct. 2001.

[Ayyagari 2000] D. Ayyagari, A. Michail, and A. Ephremides**,** *A Unified Approach to Scheduling, Access Control and Routing for Ad-Hoc Wireless Networks*, in Proceedings of the Vehicular Technology Conference (VTC 2000-Spring), pp. 380 –384, Volume 1, 2000.

[Bagrodia 2000] R. Bagrodia, M. Gerla, J. Hsu, W. Su, and S.-J. Lee, *A Performance Comparison Study of Ad Hoc Wireless Multicast Protocols*, in Proc. of the Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM) , pp. 565 –574, March 2000, Volume 2.

[Barrett 2002] Chris Barrett, Achla Marathe, Madhav V. Marathe, and Martin Drozda, *Characterizing the Interaction between Routing and MAC Protocols in Ad-Hoc Networks***,** in Proc. of the Third Symposium on Mobile Ad Hoc Networking and Computing, pp.92-103, Lausanne, Switzerland, June 2002.

[Broch 1998] J. Broch, D.A. Maltz, D.B. Johnson, Y.-C. Hu, and J. Jetcheva, *A Performance Comparison of Multi-Hop Wireless Ad Hoc Network Routing Protocols*, in Proc. of the Fourth Annual International Conference on Mobile Computing and Networking (MobiCom' 98), pp85–97, October 1998.

[Bruschi 2002] D. Bruschi and E. Rosti, *Secure Multicast in Wireless Networks of Mobile Hosts*, Mobile Networks and Applications, pp. 503-511, Vol. 7, No. 6, December 2002.

[Cheng 2001] E. Cheng, *On-demand Multicast Routing in Mobile Ad Hoc Networks*, M.Eng. Thesis, Carleton University, Ontario, Canada. January 2001.

[Chiang 1997a] C.-C. Chiang, M. Gerla, and L. Zhang, *Shared Tree Wireless Network Multicast*, in Proc. of the Sixth International Conference on Computer Communications and Networks, pp. 28-33, 1997.

[Chiang 1997b] C.-C. Chiang and M. Gerla, *Routing and Multicast in Multihop, Mobile Wireless Networks*, in Proc. of the IEEE International Conference on Universal Personal Communications (ICUPC' 97), pp. 28-33, 1997.

[Chiang 1998a] C.-C. Chiang, M. Gerla, and L. Zhang, *Adaptive Shared Tree Multicast in Mobile Wireless Networks*, in Proc. of Globecom '98, pp. 193 -207, Sydney, Australia, November 1998.

[Chiang 1998b] C.-C. Chiang, *Wireless Network Multicasting*, PhD dissertation, University of California, Los Angeles, Department of Computing Science, 1998.

[Chiang 1999] C.-C. Chiang, M. Gerla, and L. Zhang, *Tree Multicast strategies in Mobile, Multihop Wireless Networks*, ACM/Baltzer Journal on Mobile Networks and Applications (MONET), pp. 193-207, Vol. 4, No. 3, 1999.

[Chumchu 2002] P. Chumchu and A. Seneviratne, *HMoM: A Reliable Mobile Multicast Scheme Using Hybrid ARQ: Proposed Architecture and Performance Analysis*, in Proc. of Wireless 2002, pp. 464-473, Alberta, Canada, July 2002.

[Das 2001] S.R. Das, C.E. Perkins, E.M. Royer and M.K. Marina, *Performance Comparison of Two On-demand Routing Protocols for Ad hoc Networks*, IEEE Personal Communications Magazine special issue on Ad hoc Networking, pp. 16-28, February 2001.

[Das 2002] S.K. Das, B.S. Manoj, and C.S.R. Murthy, *A Dynamic Core Based Multicast Routing Protocol for Ad hoc Wireless Networks*, in Proc. of the Third Symposium on Mobile Ad Hoc Networking and Computing, pp. 24-35, Lausanne, Switzerland, June 2002.

[Ding 2002] W. Ding, *Multicast Routing in Fixed Infrastructure and Mobile Ad Hoc Wireless Networks with a Multicast Gateway*, M.Sc. (ISS) Thesis, Carleton University, Ontario, Canada, July 2002.

[Fang 2002] Yue Fang and A.B. McDonald, *Cross-Layer Performance Effects of Path Coupling in Wireless Ad Hoc Networks: Power and Throughput Implications of IEEE 802.11 MAC*, in Proceedings of the IEEE Performance, Computing, and Communications Conference, pp. 281-290, 2002.

[Fidge 1988] C. Fidge, *Logical Time in Distributed Computing Systems*, IEEE Computer, pp. 28-33, Vol. 24, No. 8, August 1991.

[Garcia-Luna-Aceves 1999] J.J. Garcia-Luna-Aceves and E.L. Madruga, *The Core-Assisted Mesh Protocol*, IEEE Journal on Selected Areas in Communications, pp. 1380-1394, vol. 17, no. 8, Aug. 1999.

[Gopalsamy 2002] T. Gopalsamy, M. Singhal, D. Panda, and P. Sadayappen, *A Reliable Multicast Algorithm for Mobile Ad Hoc Networks*, in Proc. of the Distributed Computing Systems Workshops, pp. 563-570, Vienna, Austria, July 2002.

[Grudin 2002] J. Grudin, *Group Dynamics and Ubiquitous Computing*, Communications of the ACM, pp. 74-78, Vol. 45, No. 12, December 2002.

[INRIA 2003] Adaptive EDCF NS2 code, http://www-sop.inria.fr/planete/qni/Research/AEDCF/

[Jetcheva 2001] J.G. Jetcheva and D.B. Johnson, *Adaptive Demand-Driven Multicast Routing in Multi-Hop Wireless Ad Hoc Networks*, in Proc. of the 2001 ACM International Symposium on Mobile Ad Hoc Networking & Computing (MobiHoc 2001), pp. 33-44, ACM, Long Beach, CA, October, 2001.

[Jiang 2002] M. -Y. Jiang and W. Liao, *Family ACK Tree (FAT): A New Reliable Multicast Protocol for Mobile Ad Hoc Networks*, in Proc. of the IEEE Int. Conf. on Communications, pp. 3393-3397, NY, USA, April 2002.

[Johnson 2001] D.B. Johnson, D.A. Maltz, and J. Broch, *DSR: The Dynamic Source Routing Protocol for Multi-Hop Wireless Ad Hoc Networks*, in Ad Hoc Networking, edited by Charles E. Perkins, Chapter 5, pp. 139-172, Addison-Wesley, 2001.

[Kanodia 2001] V. Kanodia, C. Li, A. Sabharwal, B. Sadeghi, and E. Knightly, *Distributed Multi-hop Scheduling and Medium Access with Delay and Throughput Constraints*, in Proc. of the 7th Conf. on Mobile Computing and Networking, pp. 200-209, Rome, Italy, July 2001.

[Kunz 2002] T. Kunz, *Multicasting: From Fixed Networks to Ad Hoc Networks*, in Handbook of Wireless Networks and Mobile Computing, pp. 495-507, John Wiley & Sons 2002, ISBN 0-471-41902-8.

[Kuri 2001] J. Kuri and S.K. Kasera, *Reliable Multicast in Multi-Access Wireless LANs*, Wireless Networks, pp. 359-369, Vol. 7, 2001.

[Lamport 1978] L. Lamport, *Time, Clocks and the Ordering of Events in a Distributed System*, Communications of the ACM, pp. 558-565, Vol. 21, No. 7, July 1978.

[Lee 1999] S.-J. Lee, M. Gerla, and C.-C. Chiang, *On-Demand Multicast Routing Protocol*, in Proc. of the IEEE Wireless Communications and Networking Conference (WCNC ' 99), pp. 1298-1304, Sept. 1999.

[Lee 2000] S.-J. Lee, W. Su, J. Hsu, M. Gerla, and R. Bagrodia, *A Performance Comparison Study of Ad Hoc Wireless Multicast Protocols*, in Proc. of the 19th Conference of the IEEE Computer and Communications Societies (IEEE INFOCOM 2000), Vol. 2, pp. 565-574, Tel Aviv, Israel, Mar. 2000.

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

[Li 2001] J. Li, C. Blake, D. De Couto, H.I. Lee, and R. Morris, *Capacity of Ad Hoc Wireless Networks*, in Proc. of the 7th Conf. on Mobile Computing and Networking, pp. 61-69, Rome, Italy, July 2001

[Li 2002] V. Li and Z. Zhang, *Internet Multicast Routing and Transport Control Protocols*, in Proc. of the IEEE, pp. 360-391, Vol. 90, No. 3, March 2002.

[Lim 2000] H. Lim and C. Kim, *Multicast Tree Construction and Flooding in Wireless Ad Hoc Networks*, in Proc. of the 3rd ACM International Workshop on Modeling, Analysis and Simulation of Wireless and Mobile Systems, pp. 61-68, Boston, USA, August 2000.

[Lou 2002] W. Lou and J. Wu, *On Reducing Broadcast Redundancy in Ad Hoc Wireless Networks*, IEEE Transactions on Mobile Computing, pp. 111-122, Vol. 1, No. 2, April 2002.

[Mattern 1989] F. Mattern, *Virtual Time and Global States of Distributed Systems*, in Proc. of the Workshop on Parallel and Distributed Algorithms, pp. 215-226, North Holland / Elsevier, 1989.

[Monarch 2003] The Monarch Project Software Distributions, http://www.monarch.cs.rice.edu/software.html

[Ni 2002] Qiang Ni, Lamia Romdhani, Thierry Turletti and Imad Aad. *QoS Issues and Enhancements for IEEE 802.11 Wireless LAN*. INRIA Research Report No. 4612, Nov. 2002.

[Obrzcka 1998] K. Obraczka, *Multicast Transport Protocols: A Survey and Taxonomy*, IEEE Communications Magazine, pp. 94-102, January 1998.

[Obraczka 2001a] K. Obraczka, G. Tsudik, and K. Viswanath, *Pushing the Limits of Multicast in Ad Hoc Networks*, in Proc. of the 21st Int. Conference on Distributed Computing Systems, pp. 719-722, Phoenix, USA, 2001.

[Obraczka 2001b] K. Obraczka, K. Viswanat, and G. Tsudik, *Flooding for Reliable Multicast in Multi-Hop Ad Hoc Networks*, Wireless Networks, pp. 627-634, Vol. 7, No. 6, 2001.

[Omar 2002] H. Omar, T. Saadawi, and M. Lee, *An Integrated Platform for Reliable Multicast Support in the Regional Mobile-IP Environment*, Mobile Computing and Communications Review, pp. 37-54, Vol. 6, No. 2, April 2002.

[Paul 1998] S. Paul, Multicasting on the Internet and its Applications, Kluwer Academic Publishers, ISBN 0792382005, June 1998.

[Perkins 1999] C.E. Perkins and E.M. Royer, *Ad hoc On-Demand Distance Vector Routing*, in Proc. of the 2nd IEEE Workshop on Mobile Computing Systems and Applications, pp. 90-100, New Orleans, LA, February 1999.

[RFC 1112] S. Deering, *Host Extensions for IP* Multicasting, (Status: STANDARD)

[RFC 1700] J. Reynolds, J. Postel, *Assigned Numbers*, October 1994. (Status: HISTORIC)

[RFC 2357] A. Mankin, A. Romanow, S. Bradner, V. Paxson, *IETF Criteria for Evaluating Reliable Multicast Transport and Application Protocols*, June 1998. (Status: INFORMATIONAL)

[RFC 2375] R. Hinden, S. Deering, *IPv6 Multicast Address Assignments*, July 1998. (Status: INFORMATIONAL)

[RFC 2908] D. Thaler, M. Handley, D. Estrin, *The Internet Multicast Address Allocation Architecture*, September 2000. (Status: INFORMATIONAL)

[RFC 3170] B. Quinn, K. Almeroth, *IP Multicast Applications: Challenges and Solutions*, September 2001. (Status: INFORMATIONAL)

[RFC 3171] Z. Albanna, K. Almeroth, D. Meyer, M. Schipper, *IANA Guidelines for IPv4 Multicast Address Assignments*, August 2001. (Status: BEST CURRENT PRACTICE)

[RFC 3306] B. Haberman, D. Thaler, *Unicast-Prefix-based IPv6 Multicast Addresses*, August 2002. (Status: PROPOSED STANDARD)

[RFC 3307] B. Haberman, *Allocation Guidelines for IPv6 Multicast Addresses*, August 2002. (Status: PROPOSED STANDARD)

[Romdhani 2002] Lamia Romdhani, Qiang Ni, and Thierry Turletti. *AEDCF: Enhanced Service Differentiation for IEEE 802.11 Wireless Ad-Hoc Networks*, INRIA Research Report No. 4544, 2002.

[Romdhani 2003] Lamia Romdhani, Qiang Ni, and Thierry Turletti. *Adaptive EDCF: Enhanced Service Differentiation for IEEE 802.11 Wireless Ad Hoc Networks*. IEEE WCNC' 03 (Wireless Communications and Networking Conference), New Orleans, Louisiana, March 2003.

[Rozovsky 2001] R. Rozovsky and P. R. Kumar, *SEEDEX: A MAC Protocol for Ad Hoc Networks*, in Proc. of the Second Symposium on Mobile Ad Hoc Networking and Computing, pp. 67-74, Long Beach, USA, October 2001.

[Royer 1999] E. Royer, and C. E. Perkins, *Multicast Operation of the Ad-Hoc On-Demand Distance Vector Routing Rrotocol*, in Proc. of the 5th Conf. on Mobile Computing and Networking, pp. 207-218, Seattle, USA, August 1999.

[Salzer 1984] J. Salzer, D. Reed and D. Clark, *End-to-end Arguments in System Design*, ACM Transactions on Computer Systems, pp. 277-288, November 1984.

[Sankarasubramaniam 2003] Y. Sankarasubramaniam, Ö. B. Akan, I. F. Akyildiz, *ESRT: Event-to-Sink Reliable Transport in Wireless Sensor Networks*, in Proc. of the 4th International Symposium on Mobile Ad Hoc Networking and Computing, pp.177-188, Annapolis, USA, June 2002.

[Shu 2002] L. Shu and D. Poppe, A*ssuring Message Delivery in Mobile Ad Hoc Networks with Packet Erasure Recover*y, in Proc. of the Distributed Computing Systems Workshops, pp. 14-19, Vienna, Austria, July 2002.

[Takai 2001] Mineo Takai, Jay Martin, and Rajive Bagrodia, *Effects of Wireless Physical Layer Modeling in Mobile Ad Hoc Networks***,** in Proc. of the 2001 ACM International Symposium on Mobile Ad Hoc Networking & Computing (MobiHoc 2001), pp. 87-94, Long Beach, USA, October, 2001.

[Tang 2002] K. Tang, K. Obraczka, S.-J. Lee, and M. Gerla, *A Reliable, Congestion-Controlled Multicast Transport Protocol in Multimedia Multi-hop Networks*, in Proc. of the 5th International Symposium on Wireless Personal Multimedia Communications, pp. 252-256, Honolulu, USA, October 2002.

[Varshney 2002] U. Varshney, *Multicast over Wireless Networks*, Communications of the ACM, pp. 31-37, Vol. 45, No. 12, December 2002.

[Williams 2002] B. Williams and T. Camp, *Comparison of Broadcast Techniques for Mobile Ad Hoc Networks*, in Proc. of the Third Symposium on Mobile Ad Hoc Networking and Computing, pp.194-205, Lausanne, Switzerland, June 2002.

[Xie 2002] J. Xie, R. Talpade, A. Mcauly, and M. Liu, *AMRoute: Ad Hoc Multicast Routing Protocol*, Mobile Networks and Applications, pp. 429-439, Vol. 7, No. 6, December 2002.

[Yang 2002] X. Yang and N. H. Vaidya, *Priority Scheduling in Wireless Ad Hoc Networks*, in Proc. of the Third Symposium on Mobile Ad Hoc Networking and Computing, pp. 71-79, Lausanne, Switzerland, June 2002.

[Yoon 2002] W. Yoon, D. Lee, H.Y. Youn, and S.J. Koh, *Throughput Analysis of Tree-based Protocols for Many-to-many Reliable Multicast*, in Proc. of the IEEE Int. Conf. on Communications, pp. 2523-2527, NY, USA, April 2002.

[Zhu 2002] Y. Zhu, *Pro-Active Connection Maintenance in AODV and MAODV*, M.Sc. (ISS) Thesis, Carleton University, Ontario, Canada, August 2002.

[Zhuang 2001] S.Q. Zhuang, B.Y. Zhao, A.D. Joseph, R.H. Katz, and J.D. Kubiatowicz, *Bayeaux: An Architecture for Scalable and Fault-tolerant Wide-are Data Dissemination*, in Proc. of the Int. Workshop on Network and Operating System Support for Digital Audio and Video, pp. 11-20, Port Jefferson, USA, June 2001.

# List of symbols/abbreviations/acronyms/initialisms

| | |
|---|---|
| DND | Department of National Defence |
| ACK | Acknowledgement |
| ADMR | Adaptive Demand-driven Multicast Routing |
| AEDCF | Adaptive Enhanced Distributed Coordination Function |
| AODV | Ad-hoc On-demand Distance Vector |
| ARQ | Automatic Repeat reQuest |
| BCAST | BroadCAST |
| BSR | Bytes Send Ratio |
| CBR | Constant Bit Rate |
| CTS | Clear To Send |
| DCF | Distributed Coordination Function |
| DRDC | Defence Research and Development Canada |
| DSDV | Destination Sequenced Distance Vector |
| DSR | Dynamic Source Routing |
| EDCF | Enhanced Distributed Coordination Function |
| FEC | Forward Error Correction |
| ID | IDentifier |
| IEEE | Institute of Electrical and Electronics Engineers |
| IETF | Internet Engineering Task Force |
| INRIA | Institut National de Recherche en Informatique et en Automatique |
| IP | Internet Protocol |
| IPv4 | Internet Protocol version 4 |
| IPv6 | Internet Protocol version 6 |
| MAC | Medium Access Control |

| | |
|---|---|
| MACT | Multicast ACTivation |
| MANET | Mobile Ad-hoc NETwork |
| MAODV | Multicast extensions for AODV |
| MBone | Multicast Backbone |
| Mbps | Megabits per second |
| ms | milliseconds |
| NACK | Negative ACKnowledgement |
| NS2 | Network Simulator version 2 |
| ODMRP | On-Demand Multicast Routing Protocol |
| OLSR | Optimized Link State Routing |
| PDR | Packet Delivery Ratio |
| PSR | Packet Send Ratio |
| RFC | Request For Comment |
| RREP | Route REPly |
| RREQ | Route REQuest |
| RTS | Request To Send |
| TCP | Transmission Control Protocol |
| TTL | Time-To-Live |