Computer Networks 56 (2012) 577–597

Contents lists available at SciVerse ScienceDirect

Computer Networks

journal homepage: www.elsevier.com/locate/comnet

Transparent and scalable terminal mobility for vehicular networks ☆

Gustavo Carneiro *, Pedro Fortuna, Jaime Dias, Manuel Ricardo

INESC Porto, Faculdade de Engenharia, Universidade do Porto, Portugal

ARTICLE INFO

Article history:
Received 6 October 2010
Received in revised form 5 September 2011
Accepted 17 October 2011
Available online 22 October 2011

Keywords:
Private network
Layer 2
Heterogeneous technologies
Mesh
MPLS
Vehicular networks

ABSTRACT

Future public transportation systems will provide broadband access to passengers, carrying legacy terminals with 802.11 connectivity. Passengers will be able to communicate with the Internet and with each other, while connected to 802.11 Access Points deployed in vehicles and bus stops/metro stations, and without requiring special mobility or routing protocols to run in their terminals. Existing solutions, such as 802.11s and OLSR, are not efficient and do not scale to large networks, thereby requiring the network to be segmented in many small areas, causing the terminals to change IP address when moving between areas.

This paper presents WiMetroNet, a large mesh network of mobile routers (Rbridges) operating at layer 2.5 over heterogeneous wireless technologies. This architecture contains an efficient user plane that optimizes the transport of DHCP and ARP traffic, and provides a transparent terminal mobility solution using techniques that minimize the routing overhead for large networks. We offer two techniques to reduce routing overhead associated with terminal mobility. One approach is based on TTL-limited flooding of a routing message and on the concept of forwarding packets only to the vicinity of the last known location of the terminal, and then forward the packets to a new location of the terminal. The other technique lets the network remain unaware for a very long time that the terminal has moved; only when packets arrive at the old PoA does the PoA send back a "binding update" message to the correspondent node, to correct the route for future packets for the same terminal.

Simulation and analytical results are presented, and the routing protocol is shown to scale to large networks with good user plane results, namely packet delivery rate, delay, and handover interruption time.

1. Introduction

In this paper we address a class of moving or vehicle-to-infrastructure networks. Our particular network—the WiMetroNet [1]—is a private network possibly owned by a consortium of companies which jointly operate public transportation vehicles such as buses, trams, and taxis. The scenario addressed provides secure broadband wireless access to a few thousands of vehicles and vehicle stops. We assume each vehicle (e.g. bus, tram, subway train, or taxi) will use its broadband access to operate services such as video-surveillance, video broadcast, and video/voice calls. Besides, each vehicle is expected to provide wireless access to its passengers, which carry portable and conventional equipments with standard IEEE 802.11 (WLAN) interfaces and a bare IP communications stack. Passengers may access the Internet not only from the vehicles but also from the stops while waiting for the vehicles, and are allowed to communicate between themselves; they can, for instance, exchange files, play games or establish voice and video communications using their applications.

Vehicles get a broadband wireless access by using heterogeneous wireless technologies, namely IEEE 802.16

☆ This work was funded by the Portuguese government through FCT grants SFRH/BD/23456/2005, SFRH/BD/20173/2004, and SFRH/BD/22514/2005, and was co-supported by the SitMe project from QREN – ON.2 program.

* Corresponding author.
E-mail address: gjcarneiro@gmail.com (G. Carneiro).

1389-1286/$ - see front matter © 2011 Elsevier B.V. All rights reserved.
doi:10.1016/j.comnet.2011.10.007

(WMAN, WiMax), IEEE 802.11, and 3GPP Universal Mobile Telecommunications System (UMTS). Each vehicle has a WMAN access which may not be accessible from every place, a WLAN access which is used when the vehicle approaches some stops, and an UMTS access which it uses when uncovered by the other technologies. Vehicles and vehicle stops are equipped with a communication equipment—the *Rbridge*—which manages the wireless broadband access and serves one or more WLAN Access Point (AP) located inside the vehicle, to which the passenger or other vehicle equipments can associate. Fig. 1 presents the WiMetroNet reference architecture. When, for instance, a passenger arrives to a tram station, he gets a secure wireless access and IP connectivity. While moving from the tram station to a tram, from the tram to the arrival station, and from there to a bus, the passenger is expected to maintain its connection and observe no considerable degradation on the quality of his communications.

The WiMetroNet is a mesh network of moving Rbridges. It is auto-configurable, and it operates at layer 2.5 over heterogeneous wireless technologies. A new routing protocol is proposed, and secure mechanisms for authorization, authentication and confidentiality are used. A passenger's equipment will see WiMetroNet as its LAN, thus being one IP hop away from the other terminals attached to WiMetroNet and from its default router. Because terminals are always virtually on the same LAN, while roaming they maintain their IP addresses. Moreover, depending on the mobile terminal implementation, in some cases DHCP renew may not even be necessary while roaming, only when the lease expires.

In order to take advantage of the ability to keep stable IP addresses in mobile terminals, the WiMetroNet network should be reasonably large. We envison many hundreds to a few thousands of mesh routers, some mobile, some fixed, and thousands or tens of thousands of mobile terminals. A traditional IEEE 802 based layer 2 forwarding architecture, which we try to emulate, is not suitable due to the way broadcasts are handled [2]. For instance, if each terminal sends one broadcast ARP packet per minute, for a network with ten thousand terminals this translates into $\frac{10000-1}{60} = 167$ packets per second received by every other host, leading to a virtual collapse of the network. Not only does a simple 802.1D bridged solution exhibit this problem, but also the new 802.11s wireless mesh standard. In the control plane, similar scalabilty problems exist. Adhoc routing protocols rely on the ability to flood the network to discover routes. The flooding can be periodic, in the case of proactive routing protocols like Optimized Link State Routing (OLSR [3]), or reactive as in Adhoc On-demand Distance Vector (AODV [4]), but in both cases results a considerable fraction of the network capacity to be consumed just for routing messages. Recent work on VANET routing protocols has focused on reactive adhoc routing protocols augmented with location information in order to provide better scalability. However, the position of a destination node can only be obtained by a sending node by asking a location server, with poor scalability as a result, or by position estimation based on a previous position, with inherent computational complexity and statistical error.

The main contributions of this paper are threefold: 1. a new scalable data plane that (a) solves broadcast problems by forbiding broadcasts in general and providing only DHCP and ARP support in a way that does not require flooding the entire network for each DHCP/ARP request, (b) uses an MPLS header when encapsulating user frames, so that it works with heterogenous technologies, including those that do not use IEEE 802 addresses, and (c) offers a LAN-like service model to support all 802.11-based end user mobile terminals; 2. an associated routing protocol that supports mobility of both Rbridges and mobile terminals and distributes the IP/MAC associations needed for the data plane ARP optimizations; 3. routing optimizations that allow fast dissemination of mobile terminals' location changes (fast handover) using only a residual routing overhead, but without needing location information (GPS) or any kind of network segmentation or hierarchy.

The paper is organized as follows. Section 2 shows some work related to our problem. Section 3 describes the base WiMetroNet architecture. Section 4 presents two approaches to optimize terminal mobility. In Section 5 the proposed solution is evaluated, including simulation results, analytical study, and benchmark of the key building blocks of a future implementation, to determine the

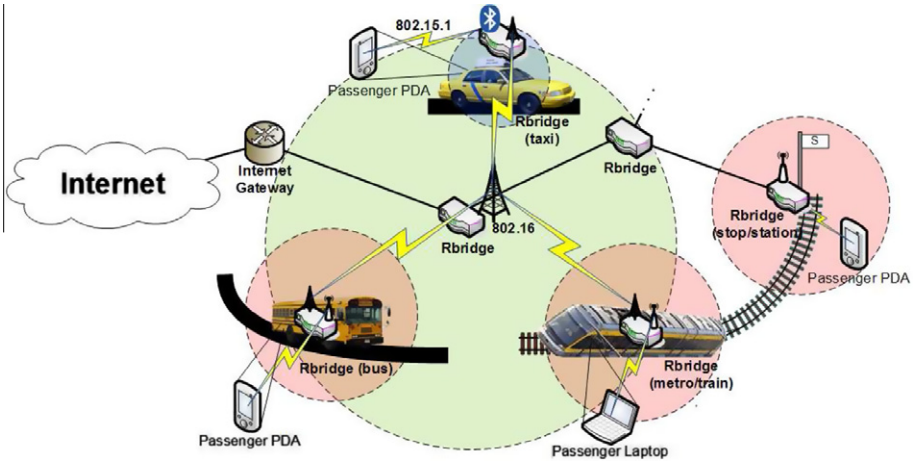


Fig. 1. The WiMetroNet reference network.

scalability limits of a WiMetroNet network. Finally, Section 6 concludes the paper with a summary and some directions for future work.

2. Related work

The IETF Transparent Interconnection of Lots of Links (TRILL) [5] is working towards a standard solution for shortest-path frame routing in a multiple-hop 802.1-compliant network with arbitrary topology. For that purpose, TRILL proposes the concept of Routing Bridge (Rbridge), a node running a link-state routing protocol at L2. Other goals of the solution are: minimal configuration, routing loop mitigation (through the use of a TTL field) and legacy node support. However, TRILL lacks efficient and scalable mobility support, which is required in the WiMetroNet scenario. Additionally, because TRILL targets perfect compatibility with the service offered by legacy bridges, it does not limit broadcasts in general, and so does not scale to large network sizes. TRILL does optimize ARP and DHCP, but there are other services that also use broadcast or multicast, such as NetBIOS, UPnP, and DNS-SD. These are not needed in a metropolitan area, but their use can take down a network, intentionally or not (''denial of service'' attacks).

The LANMAR [6] routing protocol targets large scale adhoc networks. To accomplish its scalability goals, LANMAR nodes are organized into subnets. Each node has two logical identifiers, a subnet identifier, and a node identifier unique within its subnet. Within each subnet, one of the nodes is elected as the ''landmark'' node. When a packet from one subnet targets a node in another subnet, the packet follows the path to the landmark node instead of the end node, until it enters the target subnet, at which point the path follows the most direct route to the target end node. The drawback of the LANMAR approach is that it requires prior assignment of nodes into logical subnets considering how the nodes are naturally grouped and are likely to move. The example given for LANMAR is one of a military structure, with tanks and other units orbiting the tanks. This approach does not work well when no a priori structure of the moving nodes can be defined, or when the network topology is highly dynamic. LANMAR handling of ''drifters and isolated nodes'' is complex and works well only when the fraction of drifters is small compared to the rest of the network. Our routing protocol follows a different approach, in which the entire network uses a flat addressing scheme and does not require any subdivision into different areas or groups, nor does it require any mechanism for electing a ''master'' of each group.

The Cluster-based OLSR extensions [7] are another method proposed for a highly scalable adhoc routing protocol. It assumes some *clustering mechanism* is being executed in adhoc networks, and proposes to have OLSR operate at two hierarchical levels simultaneously, intra-cluster and inter-cluster. The intra-cluster OLSR traffic does not get forwarded beyond the limits of each cluster; only the inter-cluster traffic gets globally flooded, thereby reducing control traffic overhead. The authors offer no hint on exactly which clustering mechanism is to be used, how much time it takes to converge, or what happens to the formed clusters when the topology of the network changes radically. The simulation results for Cluster OLSR shown in [7] are based on 100 nodes, and some of those nodes are static and manually selected as cluster-heads. In contrast, our routing protocol does not require election mechanisms nor any kind of hierarchy; it achieves high scalability while remaining completely flat.

The IEEE 802.11s [8] is the standard for wireless mesh networks (WMN) using WLAN interfaces. In a 802.11s WMN, nodes perform the role of Mesh Point (MP), which includes the exchange of routing messages and the frame forwarding. A node may also be a Mesh Access Point (MAP), which may require an additional standard 802.11 interface configured as an AP to provide access to legacy terminals; or/and a Mesh Portal Points (MPP), in which case it functions as a gateway to the Internet or to other non-mesh networks. The proposed routing for 802.11s is the Hybrid Wireless Mesh Protocol (HWMP), which is similar to AODV but operates at L2. It also includes tree-based routing for the WMN MPPs. Nodes initiate communication using the root-based tree node; this can create a bottleneck in the root node. Simultaneously, nodes broadcast a request for an optimized path between them, but this adds to the delay and can cause broadcast storms. User plane broadcast traffic, such as ARP and DHCP requests, exacerbates the broacast storm problem. These facts contribute to the limited scalability of the 802.11s WMNs, which is known to support well only a few tens of nodes.

Most location-aided routing protocols (LAR [9], and derivatives such as PMLAR [10], and others [11,12]) are reactive by nature and rely on the existence of location information (like GPS) in order to limit the flooding scope of route request messages. These routing protocols require a source node to be able to discover the location of a destination node. In some cases, a *location server* is employed, but clearly the communication with this server is an additional source of delay for route discovery (in addition to the route request/route reply pair), and limits scalability considerably due to its centralized architecture. In other cases, the position of a node is estimated based on the last known position, velocity vector, and the application of complex statistical models. The latter approach has two main problems. First, initially there is no ''last position'' known for any given node, and so the source node has to revert to classic flooding, which limits scalability. Second, the statistics involved are computationally intensive, and can potentially become a bottleneck, depending on router processing power and size of network; a router to put inside a vehicle must be small and not very powerful, and the network tends to be large. Finally, the assumption that location information is always available all the time may not hold if we consider routing in vehicles that spend considerable time underground, as is the case of subway trains.

The MAMP [13] protocol adds support for localized mobility management in mesh networks, achieving low handoff delay through the use of multi-path routing. However, although this solution is mostly network based, it requires some special signalling with the mobile terminals, and therefore does not work with legacy terminals.

MobiMESH [14] is a wireless mesh network organized in a core area, responsible for the mesh routing and mobility management, and in an access area that supports legacy 802.11 terminals. The routing is performed using the OLSR protocol, and therefore limited to its scalability. To signal terminal mobility events, the authors propose the use of OLSR HNA messages, which are flooded through the core, and will not scale in a scenario with thousands of mesh nodes.

In [15] the authors propose a novel network-based local mobility management scheme called "Ant", which requires only network-side changes and manages mobility transparently for legacy terminals. It achieves very good handover interruption times, and it is efficient for both intra-domain communications and access to the Internet. As a downside, the Ant scheme relies on a Location Server, which needs to be often contacted, including once for every mobile host handover. This scheme therefore does not scale to tens of thousands of mobile hosts.

The SMesh [16] network supports fast handover of legacy 802.11 terminals connected to a mesh network, by allowing more than one access point to serve a wireless client. They both monitor the link quality, and the best link is selected. Unfortunately this requires the clients to be configured in 802.11 ad hoc mode, which limits the available capacity of the network, due to requirement of a single-channel setup when in ad hoc mode.

The Enhanced Mobility Management (EMM) proposal [17] for WMNs "manages mobility without the need, for end-users, to install any software or modify their protocol stack". It works with the Neighbor Discovery protocol of IPv6, which makes it an IPv6-specific solution. Additionally, it uses multicast request messages (MCREQ) to find out the location of terminals, which does not scale for large networks.

FastM [18] is an improvement of EMM that optimizes the case of handover of a terminal between two adjacent mesh routers. Due to the dynamic nature of the vehicular scenario, this mechanism may fail if handover occurs between two nodes that have not yet realized they are neighbors, for instance, a passenger switching from bus stop to a newly arrived bus.

3. The WiMetroNet base architecture

3.1. Overview

The WiMetroNet network, exemplified in Fig. 1, is generally structured in the following way. There are Rbridges in vehicles and bus stops or tram stations. They provide 802.11 connectivity to some vehicle equipments and to the users' terminals. Vehicles connect to the network core Rbridges through 802.16, while moving, or through 802.11 to the bus or tram stops Rbridges', while stationed near them; the Rbridges in bus stops or metro stations are connected to the core via high speed wired links, where possible, or fixed 802.16a wireless connections, for the most remote locations. At the WiMetroNet core a number of Rbridges are deployed in a Gigabit Ethernet mesh topology, and the WiMetroNet control plane ensures that optimum paths are used for forwarding traffic between different edge Rbridges. Finally, there is at least one IP router functioning as Internet gateway.

The terminals connect to one of the edge Rbridges and acquire an IP address through DHCP (the DHCP broadcast requests are tunneled to a well known DHCP server). The user traffic is then encapsulated when entering the WiMetroNet network, transported inside the network, and the original frames delivered to the destination station, or to the Internet gateway. Due to scalability concerns, in WiMetroNet broadcasts are strictly controlled: although single-hop[1] broadcasts work as expected, multi-hop broadcasts are forbidden by default, as recommended in [2]. Special algorithms and optimizations have to be employed for ARP, DHCP, and generic multi-hop service discovery. Unlike L3 adhoc networks, which use routing protocols such as OLSR and AODV, the WiMetroNet architecture does not require terminals to run any kind of special protocol, routing or otherwise; only the standard 802.11 family of protocols.

Because we want to support heterogeneous L2 technologies (typically, 802.11, 802.16, and 802.3), a new L2.5 header was introduced. Since the Multi-Procol Label System (MPLS, RFC 3031) header is well known and fulfils our requirements, it was adopted for the WiMetroNet user plane. This MPLS header is very simple: it contains a 20-bit integer, for the "label", 3 bits marked "experimental" that are often used for QoS, one "bottom-of-stack" bit (for stacked MPLS headers), and an 8-bit TTL counter, totalling 32 bits. Using a standard MPLS header has the advantage of potentially allowing us to take advantage of MPLS switching hardware that already exists. On top of the MPLS layer the original L2 frames, from end terminals, are encapsulated. The WiMetroNet routing protocol, WMRP, runs on Rbridges to disseminate topology information, allowing it to build MPLS paths. Also distributed by WMRP are the list of terminals (MAC identifiers) associated to each Rbridge, as well as the list of IP-MAC associations (DHCP leases).

In the WiMetroNet architecture, heterogeneous technologies are supported, and the same Rbridge may have multiple interfaces, which are usually always active, though not all used. It is the routing protocol (WMRP) and its metrics that decides which interface to use at any given time for a given destination. In WiMetroNet, preference is given to 802.11 interfaces, then 802.16, and finally UMTS, although this ordering is configurable. When, for instance, a bus loses 802.16 connectivity, it will usually already have a UMTS link active, to be used as soon as the routing agent detects the 802.16 link failure.

The MPLS based data plane works in coordination with the control plane in the following way. First, each Rbridge knows its own RID (Rbridge Identifier), which is unique[2] in the network. By default (best-effort), packets are forwarded by applying an MPLS label that is the numerically equal to the RID of the egress (destination) Rbridge. Thus, no label

[1] Up to the first Rbridge, covering e.g. a single bus.

[2] The method for determining this unique ID is outside the scope of this paper. It might be based on a hash of the MAC address of an interface, for instance.

negotiation protocol is required for the best-effort paths, only the routing protocol.

The greatest advantage of using MPLS as forwarding mechanism is that we open up the architecture to future extensions with no (or only minor) modifications to the data plane. For instance, we could add support for traffic engineered paths. We could begin by letting labels with values below 100,000 to be reserved for RIDs, and the other values would be available for dynamic label assignments. Next, we would apply a traffic engineering protocol (e.g. RSVP-TE, adapted) to reserve a new path, by using only label values greater than 100,000. From the point of view of the MPLS switching engine, there is no difference between best-effort path discovered by the routing protocol and engineered paths reserved by other means. Other examples of what can be accomplished using MPLS include VLANs (virtual LANs), alternative backup paths for resilience to failure, and multicast trees. Such improvements are possible without modifying or extending the encapsulation header used in the data plane.

3.2. Detailed description

3.2.1. Introduction

The WiMetroNet control plane revolves around WMRP, a routing protocol that is inspired by OLSR, but which diverges from it in a number of ways. Like OLSR, WMRP is a link state, adhoc routing protocol designed for mobile wireless network. Like OLSR, WMRP defines HELLO messages for link sensing (discover neighbors), and TC (Topology Control) messages for disseminating network topology among all the nodes.

Among the differences between WMRP and OLSR we may include different address formats used, and new message types defined in WMRP to handle this type of network. While OLSR uses globally unique IPv4 addresses to identify each node participating in the adhoc cloud, in WMRP each node is assigned a unique 20-bit, MPLS compatible label. Additionally, WMRP defines two additional messages to cater for the needs specific to this type of network—MC (MAC Control) and IC (IP Control)—whose purpose is explained below. Finally, WMRP does not elect MPRs for use in flooding, for reasons stated at the end of this subsection.

3.2.2. WMRP PDU format

Like in OLSR, WMRP defines a *packet header* and a *message header*, depicted in Fig. 2. Each WMRP packet may contain a number of messages. The packet header contains packet size and a sequence number, while the message header contains fields such as *message type*, *validity time* (for how long is the information valid), *message size*, *time-to-live* (number of hops the message can still be forwarded before being dropped), *hop count* (number of hops it has been forwarded already), *originator id* (number of the node that generated the message originally). The *logical clock* field is used for partial event ordering, and follows Leslie Lamport algorithm [19]. The payload of the message is to be interpreted according to the message type. Four message types are defined: HELLO, TC, MC, and IC.

The HELLO message is used for *link sensing*, i.e. to allow nodes to be discovered by their neighbors. HELLO messages are broadcast periodically (2 s, by default) by each node, but are never forwarded.

The Traffic Control (TC) message is used by each node to advertise to the rest of the network the list of links to neighbors it has discovered, along with "costs" associated with those links. The contents of a TC message is a vector of 32-bit fields; 20 of those bits represent the node id of a neighbor that has been found, while the remaining 12 bits store the link cost (or metric). The neighbor node IDs are discovered by listening to HELLO messages, while the link cost is a linear combination of factors such as bandwidth, delay, link usage monetary cost, and stability. The TC messages are generated periodically by each node and retransmitted by other nodes, after duplicates are eliminated, until the message reaches every Rbridge in the network.

The MAC Control (MC) message is similar in purpose to TC, but instead of advertising other WMRP-enabled nodes (Rbridges) it advertises a list of attached end-user terminals, each terminal represented by its MAC identifier (EUI-48). Like TC, MC messages are periodically generated and forwarded by all the other nodes.

The IP Control (IC) message is used to disseminate IP $\leftrightarrow$ MAC associations. Typically, IC messages are generated only by Rbridges directly attached to a DHCP server, using the information contained in DHCP leases.

3.2.3. WiMetroNet Rbridge system architecture

We can see in Fig. 3 a schematic of the system architecture of a typical WiMetroNet Rbridge, including control plane, user plane, and the interactions between them.

Near the top-left of the diagram, the process of periodic generation of HELLO messages by the control plane is represented. The same control plane (i.e. the WMRP routing agent) may receive HELLO messages from immediate neighbors, and fill the *Neighbor Set* with a list of neighbors discovered so far. The Neighbor Set is consulted whenever a new TC message is periodically generated, while received TC messages feed information directly into a data structure called *Topology Set*. The Topology Set contains, for each node participating in the adhoc network, a list of links, with cost, to its neighbors, thus forming a directed graph. This graph is fed into a Dijkstra Shortest Path algorithm, finally yielding information to be stored in a *Label Switching Table*. Periodically, MC messages are generated containing a list of attached terminals. The attached terminals are stored in a data structure called *Local Terminal Associations*, which can be filled by a method similar to 802.1D Learning Bridge, i.e. by inspecting the source address of incoming frames from terminals, or by L2 information, such as the list of associated 802.11 stations, in case of local 802.11 interfaces in AP mode. On the other hand, MC messages received from remote nodes will supply information to fill the data structure *Remote Terminal Associations*. Thus, by combining the information from Local and Remote Terminal Associations it is possible to find out, for each terminal MAC identifier, its current location, be it a local interface or a remote Rbridge. Some Rbridges may also periodically generate IC messages, using information supplied by a lo-

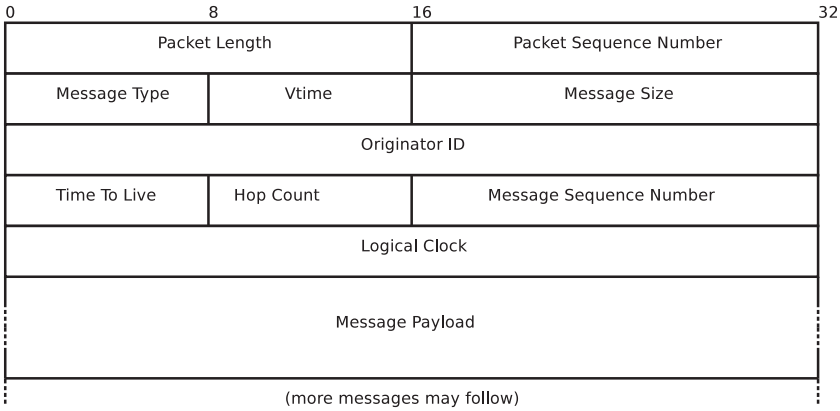


Fig. 2. WMRP PDU format.

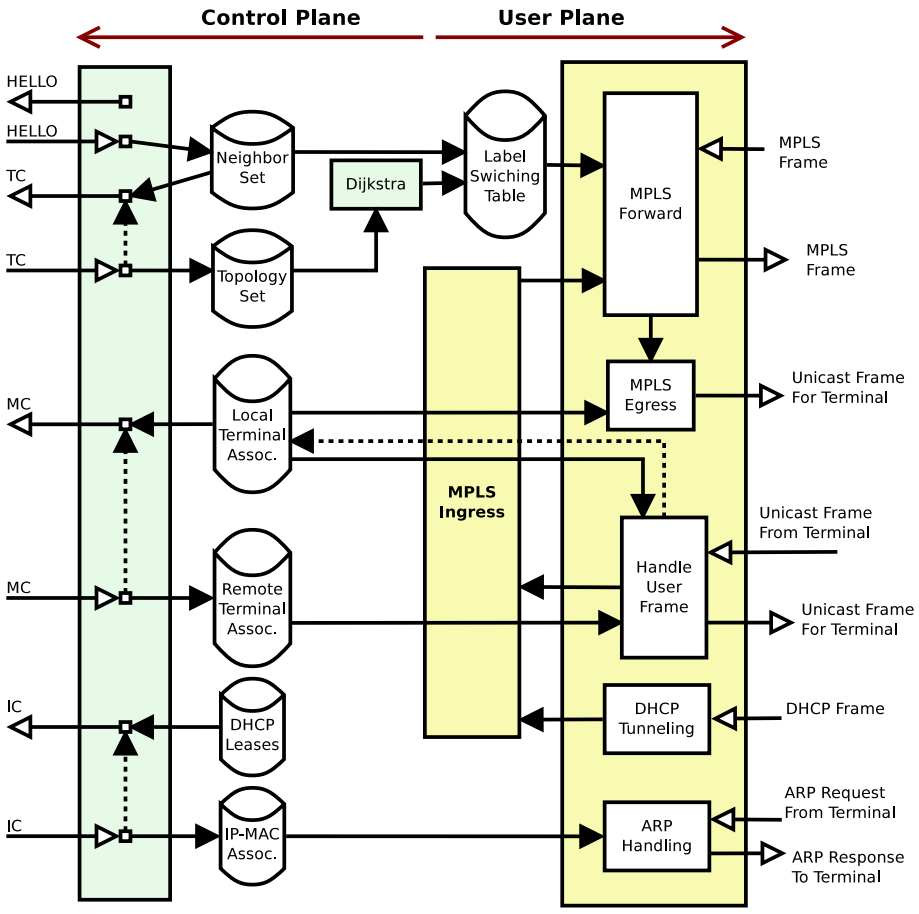


Fig. 3. Interaction between control plane and user plane in a Rbridge.

cal DHCP server. Other Rbridges will receive the flooded IC messages and fill the *IP-MAC Associations* table.

The main user plane operations are also represented in Fig. 3, towards the right of the diagram. When an MPLS-encapsulated frame is received from a core-network interface, an MPLS forwarding operation takes place, using the information in the Label Switching Table. Then the frame may either be retransmitted, remaining in the same MPLS tunnel (label switching), or it may be delivered to an attached terminal (egress). In the latter case, the Local Terminal Associations is also consulted to ascertain which of the local network interfaces the terminal is attached to. When a user data unicast frame from an attached terminal is received, the destination MAC is looked up in both Local and

Remote Terminal Associations to determine how to handle the frame. If it is for a local terminal, the frame is simply retransmitted via the network interface of the destination terminal. If, on the other hand, the destination terminal is instead found in the Remote Terminal Associations, the L2 frame enters the MPLS Ingress function, to be encapsulated and subsequently forwarded by the MPLS engine. Incoming DHCP frames are tunneled to a well known DHCP server, if the terminal is not yet know by the network, or the local Rbridge directly replies to DHCP requests, in case the terminal already known (from previous IC messages). Finally, ARP requests are intercepted by the Rbridge and an appropriate ARP reply is generated based on the information contained in the IP-MAC Associations table.

3.2.4. DHCP and terminal mobility

In the WiMetroNet architecture, all Rbridges are built to handle DHCP traffic by tunneling it to a well known DHCP server, and the reply tunneled back. End user mobile terminals are expected to acquire an IP address using this method, but what happens during handover (e.g. between a bus stop and a bus) is not so well defined. Since the SSID is the same in all buses and bus stops, the handover is a Layer 2 one from the point of view of the mobile terminals. In a L2 handover, the IP address stays the same, and the terminal does not need to renew the IP address using DHCP signalling. However, for increased robustness against misconfigured networks, some devices choose to use DHCP anyway, just to make sure they are still on the same network. In WiMetroNet, this is supported—the DHCP server will simply return the same IP address for a known terminal—although it makes the handover slightly slower.[3] In practice, "*Wireless IP Phone*" kind of terminals tend to favor L2 handover without DHCP signalling (for obvious reasons), while laptops tend to favor L2 handover with DHCP.

3.3. WiMetroNet: OLSR vs WMRP

The OLSR protocol was used as basis for WMRP, but significant modifications are done for better efficiency in the WiMetroNet scenario. The most significant difference is that WMRP operates at L2 instead of L3, which simplifies terminal mobility. Whereas with WMRP, a terminal may change point of attachment without breaking IP communications, with OLSR it is not so simple. Each OLSR router could tunnel DHCP requests to the central DHCP server, but then it needs to intercept the DHCP response and modify the "default gateway" address to be its own. To let other OLSR routers know the location of the terminal, HNA messages have to be disseminated. In WMRP, the MC messages play a role similar to HNA messages in OLSR. Thus, a WMRP protocol configured with MC refresh interval equal to OLSR's HNA default refresh interval (5 s) can be considered to have approximately the same level of routing overhead and performance as OLSR.

Another key difference is that WMRP does not inherit OLSR's Multi-Point Relay (MPR) mechanism. The MPR flooding process is used by OLSR to reduce the overhead of TC flooding to 15–40% of the classical full link state flooding, depending on the node density [20]. This overhead reduction is due to a subset of nodes, called MPRs, which are selected by other nodes to generate and forward the link state information (TCs). However, the effectiveness of MPR flooding overhead reduction is highly dependant on the average node density, which usually should not be so high due to mobile WiMax range and capacity limitations. There are also computational complexity issues. Finding an optimal MPR set is an NP-complete problem [21], and even the heuristic included in OLSR is computationally intensive. The use of MPRs also increases the time needed for the routing protocol to adapt to node mobility [22], and can decrease the robustness of the flooding process in the presence of link failure [23].

[3] Since the terminal is already known in the target Rbridge, the DHCP request is not tunneled, and the delay amounts to just one or two round-trips in the local 802.11 link.

4. Scalable terminal mobility optimizations

4.1. Introduction

What has been described so far is the base WMRP architecture, which is a simple link state routing protocol. In this base architecture, if we copy the OLSR default settings, WMRP periodically broadcasts HELLOs every 2 s, and periodically generates a new TC and a new MC every 5 s. As for the IC messages, they may be generated with a frequency as low as once every 60 s, since they convey information (DHCP leases) that changes very slowly.

In link state routing protocols, such as OLSR and WMRP, the interval between periodic routing messages that are to be flooded through the entire network, such as TC and MC, is a crucial design setting, a tradeoff between routing control traffic overhead and convergence time with mobility. Although the 5 s interval is not a bad choice for supporting mobility, it can be easily shown that it does not scale very well to large networks. The rate P of MC messages received by a node is given by $P = \frac{N-1}{\tau}$, with N representing the number of Rbridges in the network τ the message generation interval. As the routing control traffic increases linearly (on a per link basis) with the network size, there is a limit to the network size that is reached when the amount of control traffic exceeds a reasonable fraction of the network capacity.

To overcome these limitations, we begin by lowering the rate of control traffic to one message per 60 s. This reduces the control traffic to $\frac{1}{12}$ of the normal value, but leaves the routing protocol unable to adapt to node mobility in a timely manner. On top of the periodic global routing messages, additional control messages are defined and used, in order to support mobility effectively but without flooding the entire network, thus scaling better with the network size.

In the remainder of this section two methods of supporting mobility of end user terminals in an efficient and scalable way are described. Then, some simulation results are presented and discussed. Only the routing overhead due to terminal mobility is addressed. While there are mobile Rbridges in the simulations (inside buses, which are actually moving), the routing overhead caused by them is

neither optimized nor considered in this paper; solutions for moving Rbridges have been developed, are currently under evaluation, and will be included in a future publication. To account for the routing overhead of moving Rbridges, which generate TC messages, we reserve 5% of link bandwidth; together with the 5% limit we consider in this paper for the MCs (terminal mobility), we limit the total routing overhead to 10% of the link bandwidth, therefore leaving at least 90% of the network capacity for transporting user traffic. These values are just examples of limits that seem reasonable. In this paper, if we determine that a nework can scale to N nodes with 10% of routing overhead, we can expect that the network can scale to M nodes, $M > N$, if the routing overhead limit can increase to e.g. 15%.

4.2. The "explosive updates" approach

The solution that we call "explosive updates" is based on the technique of generating an MC message with a limited TTL to notify a small region of nodes around the former point of attachment of the terminal whenever a handover occurs. We call this an "explosive update". Due to this local update mechanism, a distant correspondent Rbridge computes a path to another Rbridge that ends just inside the edge of the explosive update region around the destination Rbridge. The strategy consists in reaching the closest node that was for sure updated if the terminal moved since the last global link-state update. Packets will follow this path/tunnel, egress at the region edge, find a new more up-to-date route, and ingress again, this time on the right path. To obtain the RID of the Rbridge at the edge of the explosive update region, we take advantage of the fact that the Dijkstra Shortest Path algorithm outputs a full shortest path to the destination Rbridge, not just the next hop, in other words a list of RIDs. Determining the egress point for the packets is then a matter finding the nth RID in the list, counting from the end, n being the explosive update TTL that is configured. The two MPLS tunnels are only used for reaching terminals that are attached to a distant Rbridge; if, on the other hand, the terminal was last known to be attached to a nearby Rbridge (inside the the explosive update region), then only one MPLS tunnel is needed.

Consider as an example the scenario in Fig. 4, wherein a mobile terminal T changes point of attachment from node D to node F. The network topology consists of a grid of wireless nodes, with each node having as neighbors the nodes immediately above, below, left, right, and diagonals. A corresponding node (another terminal) is attached to node A and is sending packets to our terminal of interest T. Let us consider, for demonstration purposes, that the explosive update is limited to TTL = 1. This TTL value is just an example and is configurable.

Initially, packets take the shortest path, which is split into two contiguous MPLS tunnels, A, B, C followed by C, D. Thus, two ingress operations occur in this case, one in node A and another in node C. Node A sends the packets to egress on node C because it knows that C is the node in the shortest path from A to D that is closest to A but still within the one-hop region around D, thus guaranteed to remain informed of the latest location of the terminal. As packets arrive at C, they egress the first MPLS tunnel and ingress into the second one, C, D, finally arriving at the desired destination.

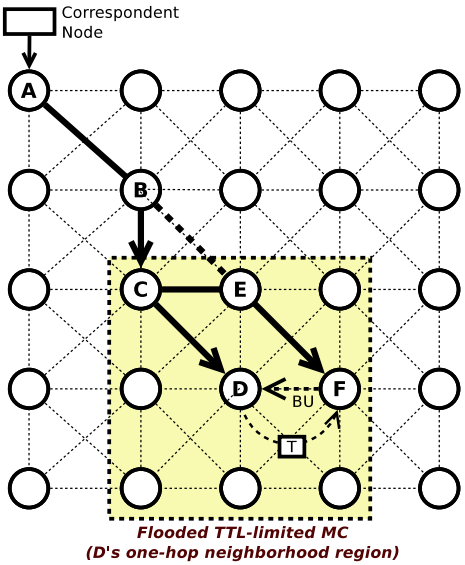


Fig. 4. The "explosive updates" terminal mobility solution.

When the handover of the mobile terminal from D to F occurs, a BU (Binding Update) message is sent from F to D, which simply notifies D that the terminal, T, is in a new location, F. Consequently, D emits a new MC message with TTL = 1, this way advertising the new location of the terminal to its one-hop neighborhood, which includes C and E. Meanwhile, A is unaware of the topology change, and keeps sending packets to node C. However, since C has been notified of the handover, it starts forwarding the packets to the correct new location. Thus, the second MPLS tunnel is no longer C, D, it has now become C, E, F.

With this solution, we have frequent updates due to mobility, but they are localized and therefore scale well for large networks. Moreover, the solution is still purely proactive, and the overhead is always the same regardless of the pattern of traffic and number of correspondent nodes. There are some drawbacks too. Even in static situations, two MPLS tunnels are used most of the time (except for nodes that are close to each other), leading to nearly twice the rate of ingress operations, which is a more expensive operation than MPLS forwarding. Additionally, during handover the routes become slightly less optimal. In the example, after the handover the optimal route would be A, B, E, F, but the actual route traversed by packets is A, B, C, E, F, which has one more hop and, consequently, larger delay. This effect will last, if the terminal does not move again, 30 s on average, until the next global MC update notifies also A that the terminal has moved.

4.3. The "binding updates" approach

This mobility optimization consists of notifying the Rbridge attached to the correspondent nodes, reactively,

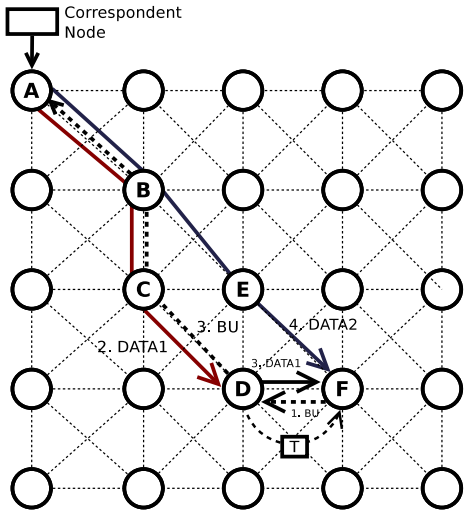


Fig. 5. The "binding updates" terminal mobility solution.

when they send packets to the old location of a node that has already moved. As an example, consider the scenario in Fig. 5. As soon as the terminal handover from node *D* to *F* occurs, a *BU* message is transmitted from *F* to *D* to notify it of the handover. However, node *A* is not notified, and keeps sending packets for the terminal towards node *D*. The first packet after the handover that *D* receives, DATA1, is forwarded by *D* to the correct new node that is currently serving as the terminal's point of attachment, *F*. Additionally, a BU message is sent by node *D* to node *A* to notify it of the new terminal location. Thus, future packets sent by *A* will, from then on, take the correct shortest path between *A* and *F*: *A*,*B*, *E*, *F*.

The "binding updates" solution is a hybrid proactive/reactive approach: it is proactive due to the slow periodic global MC updates, but is also reactive due to the BUs sent to correspondent nodes as data packets arrive at the incorrect locations. The main advantage of this solution is that packets always follow the shortest path, even after the handover, with the exception of the first packet[4] of the flow, which takes the wrong path and therefore experiences a slightly larger delay. The overhead of this approach is minimal for common scenarios. It depends mainly on the number of correspondent nodes, or, to be more precise, the number of different Rbridges attached to correspondent nodes. This works very well for common scenarios of server-client communication, especially if all the servers are behind the same Rbridge (e.g. hosts on the Internet). However, for peer-to-peer applications, with a large number of peers evenly distributed among many Rbridges, the overhead generated by the binding updates is expected to become significant.

5. Evaluation

In this section we evaluate the WMRP routing protocol, with focus on the terminal mobility optimizations that were described. For evaluation purposes, we consider that the base WMRP protocol with MC interval of 5 s to be approximately equivalent to the OLSR protocol. We consider two scenarios: a "road" scenario, representing a single bus line, and a "grid" scenario that mimics a city grid with city blocks and buses travelling between them.

5.1. Road scenario

5.1.1. Scenario

To evaluate our routing protocol, we started by a simple scenario that consists in providing coverage to a single straight bus line, with regular bus stops. This "road scenario" is not only simple enough to allow us to derive analytical expressions for the routing overhead, but also offers some realism.

It is depicted in Fig. 6, where we can see it mainly consists of a straight road, with a number (β) of equally spaced bus stops, a number ($\frac{\beta}{2}$) of equally spaced base stations, and a pyramidal topology of additional infrastructure Rbridges, interconnected with Gigabit Ethernet or fiber optic links. In our scenario, the spacing between bus stops is 1 km, as is the spacing between the row of base stations and the row of bus stops. We assume that the 802.16 base stations have a range (cell radius) of approximately 1.5 km, so that there is complete 802.16 coverage along the road. The capacity of a mobile 802.16 link varies considerably with the topographic conditions and distance to the base station, however some performance measurements put the 802.16 outdoors capacity at around 1.5–3 Mbit/s for the uplink direction, and 5–11 Mbit/s for downlink [24]. For the purpose of this study we will assume a (conservative) 802.16 capacity of 2 Mbit/s.

Along the road, a number (also β) of buses travel between bus stops. We recall that each bus contains one Rbridge, so they are the "moving network" part of the mesh network. There is a number (k) of terminals travelling inside each bus, and the same number of terminals waiting at each bus stop. The mobility model of buses is as follows. Each bus is initially stopped at a different bus stop, covering all bus stops. The buses travel from one bus stop to the next, stopping there for a period of time between 10 and 20 s, random and uniformly distributed. While stopping, all the terminals travelling inside the bus leave it and stay at the bus stop, while at the same time the terminals in the bus stop enter the bus. The buses travel between bus stops in a realistic way, by beginning with a constant acceleration period of 10 s, followed by some distance travelled at a constant speed of 22.2 m/s, and finally a constant deceleration period of 5 s, right before stopping at the next bus stop.

There is a simple server node, near the Internet gateway, that is used for traffic generation purposes. In order to measure the routing protocol convergence as effectively as possible, we should consider only packets travelling downstream, from the server to each of the terminals. In the upstream direction, even if a terminal moves to another Rbridge, the route from the new Rbridge to the server is always the same because the server does not move. In the downstream direction, however, the Rbridge near the server has to know the new location of the terminal in

[4] Or rather, the first few packets, depending on the sending rate and round-trip time between ingress and egress nodes.

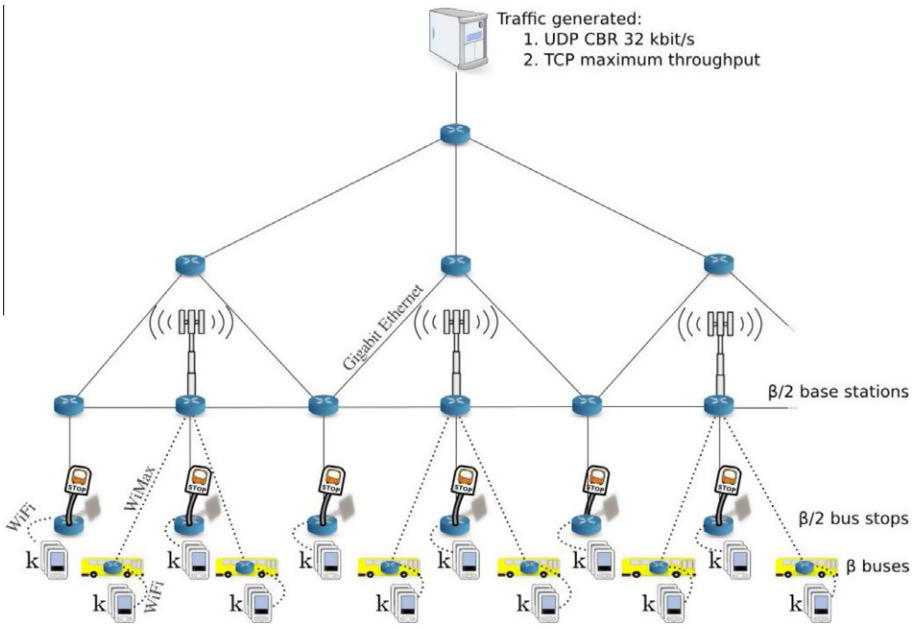


Fig. 6. The "road scenario" network topology (small network example).

order to ingress the packet into an MPLS tunnel ending in that Rbridge, and here is the true problem, here is where we need to focus our measurements.

To study the scalability of our routing protocol, we only need to increase the β parameter. The number of buses, β, is increased accordingly (1). By keeping k constant, the total number of terminals in buses and bus stops, T, also increases proportionally (2). The number of Base Stations, B, is also proportional to the number of Bus Stops (3). Likewise for the number of "tier-2" Rbridges (between base stations) (4) and "tier-3" Rbridges (5). Only the Internet Gateway and Server remain singleton as the network scales.

$$b = \beta \tag{1}$$
$$T = 2k\beta \tag{2}$$
$$B = \beta/2 \tag{3}$$
$$R_2 = \beta/2 \tag{4}$$
$$R_3 = \beta/4 \tag{5}$$

In our evaluation, we will focus on the 802.16 links of buses, which are the most resource constrained links (the remaining links being wired or WiFi) in the entire network. If routing overhead is acceptable for those links then it will necessarily be acceptable for the entire network. As previously mentioned, in this paper we only address the part of the routing protocol responsible for the mobility of terminals, namely MC messages.

5.1.2. Analytical model

The analytical model for WMRP base protocol in this scenario has already been mentioned. The rate of MC messages received on average by each node in each of its available links is given by $P = \frac{N-1}{\tau}$, with N the total number of Rbridges and τ the periodic MC refresh rate. The overhead is independent of the number of links. In this case, each MC message carries k MAC identifiers.

If μ represents the rate of MC message generation (1/MC interval), H the fixed header size of an MC message (20 bytes), and M the size of each entry inside the MC message (8 bytes), the global MC flooding overhead of the protocol can be given by (not including the mobility triggered messages yet):

$$Overhead_{global} = \mu\beta(H + (k+2)M) \tag{6}$$
$$+ \mu b(H + (k+3)M) \tag{7}$$
$$+ \mu R_2(H + 5M) \tag{8}$$
$$+ \mu B(H + 5M) \tag{9}$$
$$+ \mu R_3(H + 3M) \tag{10}$$

The overhead components can be explained by considering each type of Rbridge and the fact that even Rbridges themselves are seen as terminals by neighboring Rbridges. The first term (6) gives the overhead of bus stops, which sees k end user terminals, plus one Rbridge as terminal, plus one bus connected to it (on average). The second term (7) represents the overhead generated by buses, and it includes k terminals, plus one MC entry for another bus on average connected to the same base station, plus an MC entry for the bus stop WiFi link to which the bus is (at least part of the time) connected, and yet another entry for the base station to which it is connected. The third term (8) denotes the overhead generated by tier-2 transit Rbridges, which have 5 neighbors each. The fourth term (9) includes the overhead generated by base stations, which have 3 links each to neighboring fixed Rbridges, and on average are two more links to moving Rbridges inside buses. Finally, the term (10) expresses the overhead generated by tier-3 Rbridges, with 3 links each to report. The expression can be condensed as:

$$Overhead_{global} = \frac{\mu\beta}{4}(11H + (8k + 33)M) \quad (11)$$

For the *explosive* mobility optimization, the overhead model is similar to the base protocol overhead with $\tau = 60$; we only have to add the additional overhead caused by the BU and the TTL-limited MC flooding. It can be shown that, for *explosive* with TTL limit of 2, the additional WMRP messages that pass through the bus 802.16 links are $6k$ MCs, $2k$ BUs, and $2k$ BAs.[5] In this case, the additional mobility-related MC messages carry a single MAC identifier each.

We also need to take into account the time it takes for a bus to travel between bus stops, which is given by (16), where t_s denotes time a bus is stopped on a bus stop (20 s), t_a denotes the acceleration period (10 s), t_d the deceleration period (5 s), d_t the total distance travelled between bus stops (1000 m), d_a the distance travelled while accelerating, d_d the distance travelled while decelerating, and s the top speed of the bus (22.22 m/s, 80 km/h).

$$a = s/t_a \quad (12)$$

$$b = s/t_d \quad (13)$$

$$d_a = \frac{1}{2}at_a^2 \quad (14)$$

$$d_d = st_d - \frac{1}{2}bt_d^2 \quad (15)$$

$$t_{travel} = t_s + t_a + \frac{d_t - d_a - d_d}{s} + t_d \quad (16)$$

$$= t_s + \frac{t_a + t_d}{2} + \frac{d_t}{s} \quad (17)$$

The overhead due to handover for the *explosive* case can then be given by:

$$Overhead_{explosive} = Overhead_{global} + \frac{6kH_{MC} + 2kH_{BU} + 2kH_{BA}}{t_s + \frac{t_a+t_d}{2} + \frac{d_t}{s}} \quad (18)$$

In (18), H_{MC} represents the size of a MC message with one entry (28 bytes), H_{BU} is the size of a BU (Binding Update) message (40 bytes), and H_{BA} the size of a BA (Binding Acknowledgement) message (24 bytes). For instance, for $k = 10$, replacing the values we obtain:

$$Overhead_{explosive} = Overhead_{global} + 326.6\text{bit/s} \quad (19)$$

Clearly, the 326.6 bit/s component is constant and $Overhead_{global}$ will easily surpass it for medium-to-large networks by several orders of magnitude, thus we may make the approximation that $Overhead_{explosive} \approx Overhead_{global}$. If we were to develop an analytical expression for $Overhead_{bindupdate}$, we would draw a similar conclusion.

5.1.3. Simulation setup

For simulations, we used the NS 3.2 simulator, with additional models and back-ported bug fixes. At the link layer, the simulations were configured as follows. For 802.11, a constant data rate was set to 11 Mbit/s, rather than the adaptive one, and a simplified 802.11 MAC layer was used. The new *statistical MAC layer* simulates the 802.11 MAC service by means of a random number generator that provides transmission delays according to a log-normal probability distribution, which closely resembles the 802.11 MAC delays, congruent with what had already been concluded analytically in [25]. This technique makes our simulations run about five times faster, at the cost of a certain loss of precision. But since we are not trying to discover what is the exact capacity of the simulated network, rather just observe how the routing protocol scales, simulation precision is less important than being able to simulate larger networks. The simplified 802.11 MAC requires the definition of a hard cutoff range for the signal, which has been defined as 30 m for the AP inside buses, and 100 m for the AP in the bus stop. These are the approximate indoor and outdoor ranges, respectively, of typical WiFi 802.11b/g equipment. We also used the statistical WiFi MAC to simulate the 802.16 links, configured for a range of 1500 m and a bitrate of 2 Mbit/s. This method has less precision than the builtin NS-3 802.16 module, but allowed us to simulate much larger networks.

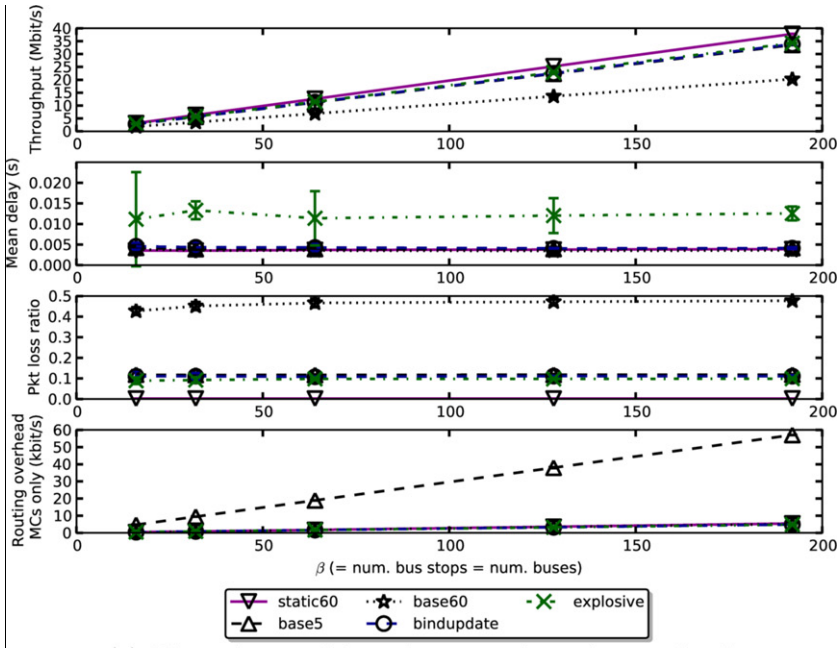
A large set of simulations were run, varying the number of bus stops between 16 and 256, and setting the k parameter to 2. Although $k = 2$ may seem limiting, it allows us to simulate larger networks. The analytical model considers other k values (see Fig. 11). Thus, the number of mobile terminals varied between 64 and 1024. Each simulation simulates 600 s, and was repeated 3 times for confidence interval purposes.[6] Additionally, at each network size the following configurations were simulated in turn:

- ***static60*** Consists in grounding all buses and mobile terminals, so that there is no mobility. The MC refresh period is set to 60 s. This simulation is used to establish the top-line of the results (the maximum network performance is expected to be attained in a static network);
- ***base5*** The base WMRP protocol, configured with OLSR settings, i.e. MC refresh period of 5 s. Mobility of both buses and terminals is present, but no terminal mobility optimizations enabled. As noted in Section 3.3, for all intents and purposes we may consider this the same as OLSR;
- ***base60*** Base WMRP with MC refresh period of 60 s;
- ***bindupdate*** The "binding updates" optimization, with mobility enabled and a 60 s MC refresh period;
- ***explosive*** The "explosive updates" optimization, with mobility enabled, 60 s MC refresh period, and TTL = 2 for the mobility MC updates.

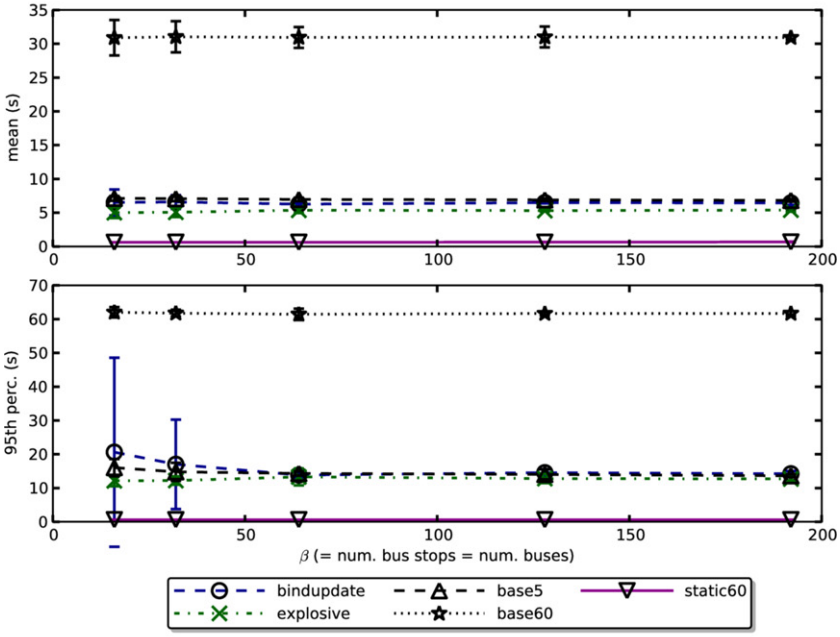
We did two sets of experiments with different traffic patterns. In one set we simulate the server transmitting a CBR UDP flow to each terminal. Each flow consists of 4 packet/s, with packet size 1000 bytes, totalling 32 kbit/s.

[5] BA stands for Binding Acknowledgement and is used to confirm the correct reception of a previous BU message.

[6] From the obtained results, due to the large number of packets processed in each simulation, we have found that repeating 3 times each simulation was enough to obtain a very small 95% confidence interval for large networks, as can be seen in the results figures.



(a) Throughputs, delays, losses, and routing overhead



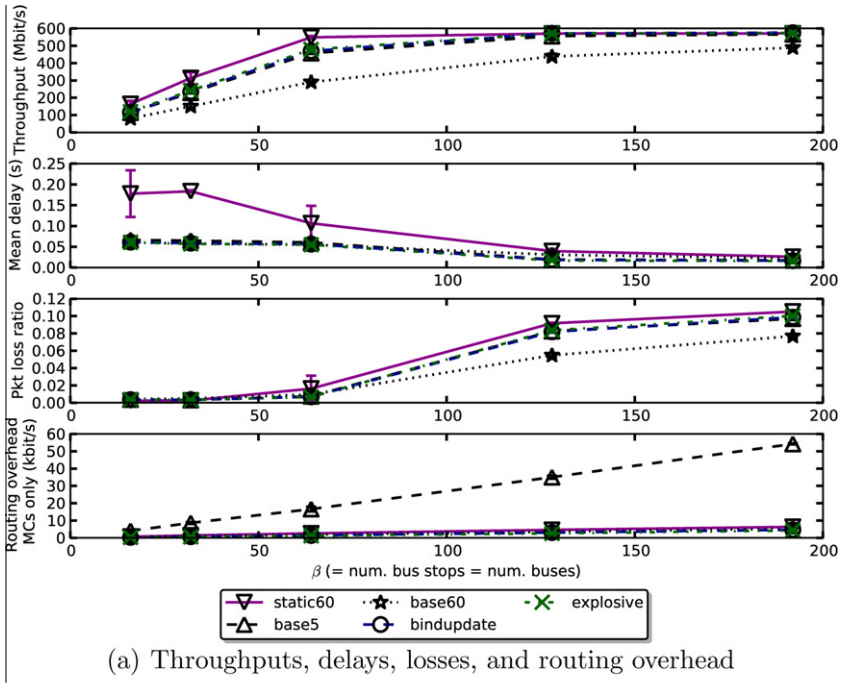
(b) Flow interruptions

Fig. 7. Road scenario: UDP results.

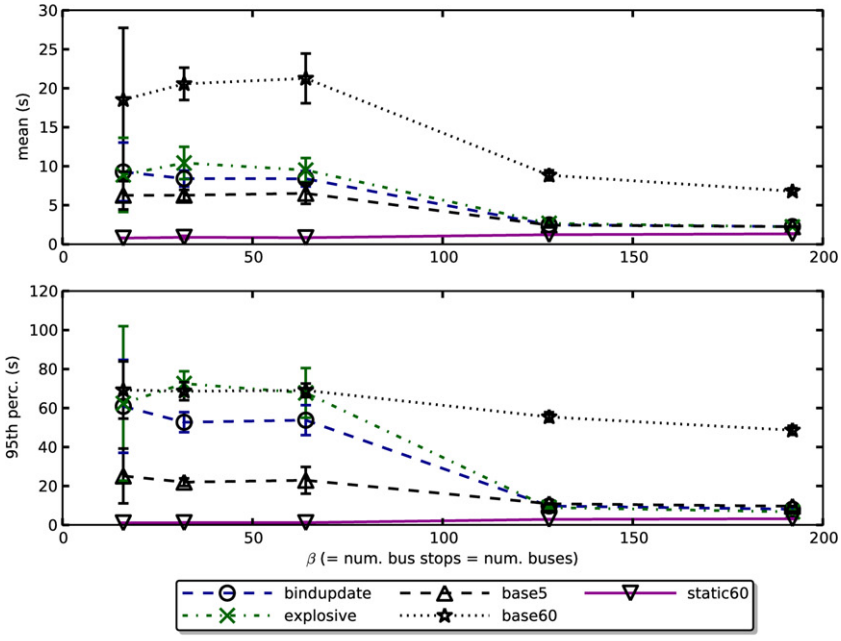
In another set of experiments there is one TCP connection between the server and each terminal, and we attempt to transmit as much data as possible through each of these connections.

5.1.4. Simulation results

The network performance results are shown in Fig. 7 (UDP) and Fig. 8 (UDP) and Fig. 8 (TCP). All the curves in all plots have represented confidence intervals, although they are too small to see in some cases. The top subplots show the received bitrate, end-to-end delays, packet losses, and routing overhead. All values are averaged over all flows, except the bitrate which is the sum of all flows, and the routing overhead which is independent of the flows and is the total bitrate that passes on the WiMax link of each bus. The bottom subplots show what we call "flow



(a) Throughputs, delays, losses, and routing overhead



(b) Flow interrutpions

Fig. 8. Road scenario: TCP results.

interruptions". We consider there is a flow interruption whenever the inter-arrival time of received packets in the terminal exceeds 0.5 s. Whenever such an interruption occurs, the duration of the interruption is recorded in a histogram. In the bottom subplots we show the mean and 95th percentile of all flow interruptions over all flows. These flow interruptions effectively measure the time take by handovers from the point of view of the applications, and is directly related to the time it takes for the routing protocol to discover new accurate routes for moving terminals.

From the presented results we may draw a number of conclusions. First, our mobility optimizations, *bindupdate* and *explosive*, achieve significantly better results, in terms of bitrate and packet loss ratio, than *base60* while generating the same routing overhead. Another comparison we

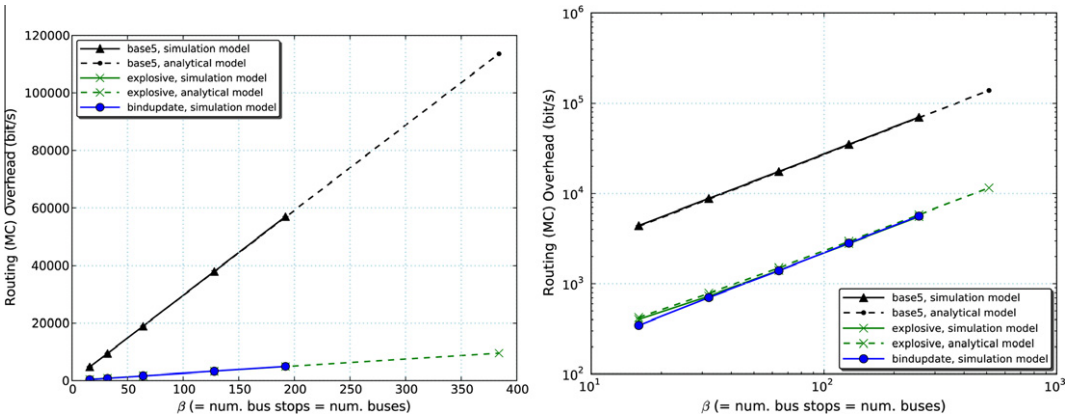


Fig. 9. Road scenario: routing overhead (MCs only) simulation and analytical results: linear scale (left) and logarithmic scale (right).

can make is that the mobility optimizations achieve identical user plane results as *base5* but using only a fraction of the control plane overhead. These conclusions hold also for the "flow interruptions" metric, and for either UDP or TCP traffic.

The explanation for these results is simple. The base60 solution has low routing overhead because it causes Rbridges to send only one MC every 60 s. But this very slow refresh period also causes routes to be frequently out of date, hence the greater packet losses and lower bitrate. On the other hand, *base5* has frequently updated routes and good user plane performance, but all the routing updates are global and so the routing overhead will be high. Our solutions have slow global updates, hence the low routing overhead, but frequent localized routing updates, which do not have global impact on the total routing overhead but which provide updated routes very quickly, hence the low routing overhead and good user plane performance.

If we would consider only the user plane results seen so far, it would seem like a simple base WMRP configuration with 5 s refresh interval (like the OLSR default TC or HNA refresh interval) is about as good a solution as the terminal mobility optimizations described in Section 4. However, this is not a complete picture. Fig. 9 shows the routing overhead obtained via simulation, together with the predicted values from the analytical models. We can see in these results that *base5* has at least an order of magnitude greater routing overhead. It can also be observed that all the mobility solutions eventually tend to a linear growth, for large networks. The non-linearity in *explosive* for small networks is due to the fixed overhead introduced due to mobility, but that fixed overhead eventually becomes insignificant with increasing network size. Finally, these results also show that the analytical and simulation models are in agreement.

5.1.5. Network scalability limit

With the results so far, the main quest for discovering the scalability limits of WMRP is yet to be completed. Via simulation we have confirmed the intuitive assumption that the routing overhead, on a per-link basis, scales linearly with the network size. To find out whether a routing protocol scales to a certain network size, ideally we would like to simulate a very large network until the routing overhead exceeds a certain limit. Unfortunately, simulating such a complex system takes a toll on computing resources and we have found it difficult to simulate networks larger than 256 bus stops/1024 mobile terminals. Instead, we will utilize analytical models to predict results for any network size without simulations. In this study, we consider 5% of the 802.16 link capacity (100 kbit/s) as an acceptable limit for the maximum bandwidth consumed by the MC messages. No packet rate limit is considered because the WMRP agent already takes care of aggregating multiple messages into a small number of large packets.

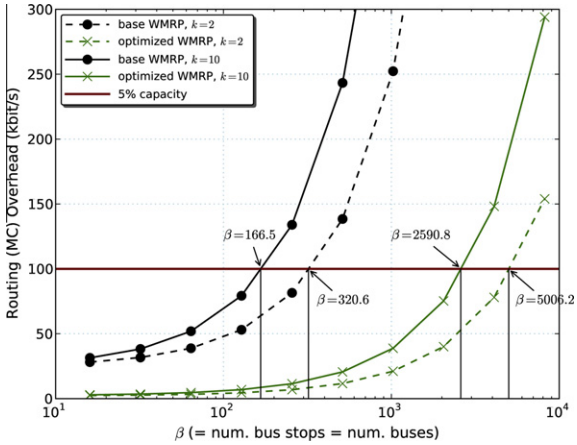


Fig. 10. Road scenario: routing protocol scalability prediction.

Fig. 10 presents the results of the scalability analysis for two values of k: 2, and 10. From these curves, we can obtain the following limits. For $k = 2$, the base protocol crosses the 5% capacity limit at $\beta = 320$ (1280 terminals[7]), but with the optimizations in effect the limit is crossed at $\beta = 5006$ (20024 terminals). For $k = 10$, the base protocol

[7] We recall that the number of terminals, T, is given by $T = 2k\beta$.

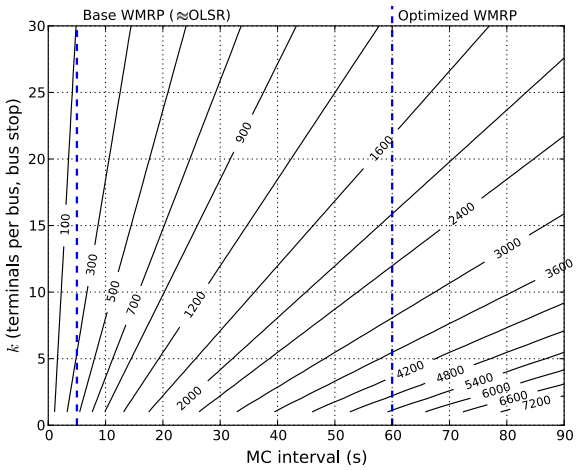


Fig. 11. Maximum β for a range of k and "MC interval" values.

crosses the 5% capacity limit at $\beta = 166$ (3320 terminals), but with the optimizations in effect the limit is crossed at $\beta = 2590$ (51800 terminals). In both cases, the optimization increases the number of bus stops and mobile terminals by a factor of approximately 16. In the case of the *explosive* mobility optimization, these limits are independent of the user traffic pattern.

According to Eq. (11), the maximum β limit is a direct function of k and μ, and μ is the inverse of the periodic MC generation interval. By replacing the overhead with the 5% limit, 100 kbit/s, and solving the equation for β as function of k and μ, we may obtain a scalar field. In Fig. 11 the evolution of the maximum β is shown for a range of k and "MC interval" values. As expected, a greater MC interval will cause less routing overhead to be generated, and so allows β to attain larger values, i.e., larger network size. However, a too large interval between global updates would bring other problems, such as the increased transient state that Rbridges have to keep due to terminal mobility. On the other hand, a larger k value (terminals per bus or bus stop) will cause each MC message to carry more MAC identifiers and so generate more routing overhead, leading to a reduced network size. The k parameter cannot be controlled, but it is not expected to be large; a large number of active terminals are expected, but they should naturally spread among many different buses or bus stops (or else even the WiFi network will be overloaded).

5.2. City grid scenario

5.2.1. Scenario

We consider the scenario of a city organized as a grid of city blocks, exemplified in Fig. 12, for the case of a very small network. Each block has the typical "Manhattan" city block size of 80 by 274 m, which is a prototypical size followed in many cities around the world. Between city blocks we consider that there exist roads for vehicles, and at each road intersection we assume the presence of a traffic stop sign. There are a number of bus stops spread throughout the city grid, at the ratio of one bus stop every 2 blocks vertically (we consider the larger side of each city block the vertical side), and one every 4 blocks horizontally.

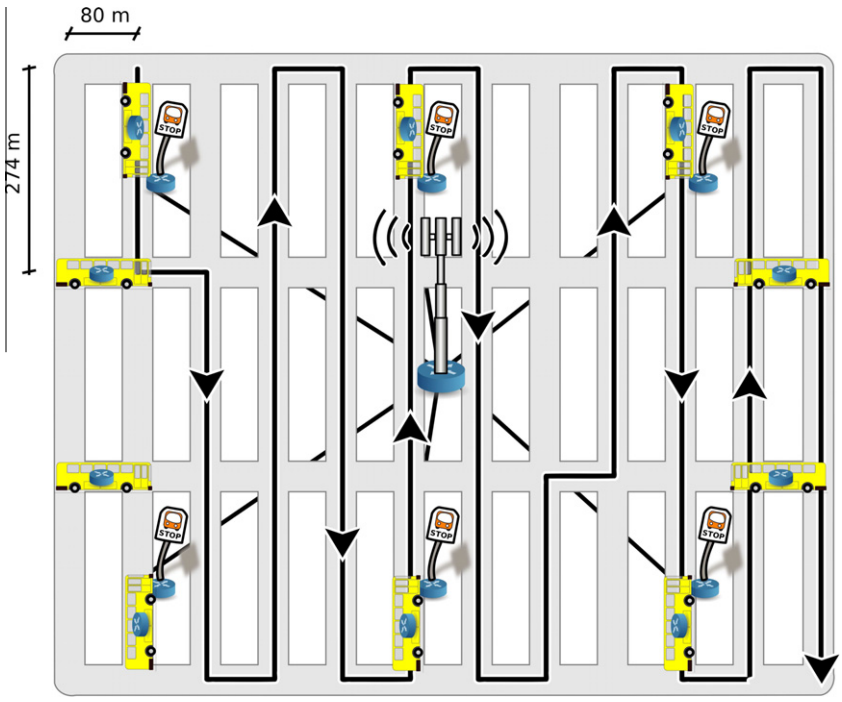


Fig. 12. Example "city grid" topology, showing some buses at their initial positions and the path of one of those buses.

At the beginning of the simulation, a number of buses exist at each of the four edges of the city grid. At each of the north and south edges we have 1/4 as many buses as the horizontal number of city blocks, while at the east and west edges we have half as many buses as the vertical number of blocks. The buses cross the city center travelling along existing roads from one edge to the other, stopping at each intersection, due to the traffic stop signs, in conformance with a Stop Sign Mobility Model [26]. They will also stop for 20 s at each bus stop along the path, allowing the passengers to enter and leave the bus. A sample path taken by a bus is shown by the solid line with arrows in Fig. 12; as seen there, the bus route is vaguely a zig-zag line, but constrained to the city roads, ensuring that the whole city area is covered.

The city area has full WiMax coverage, provided by grid of equally spaced base stations. The WiMax bandwidth and range vary greatly with the topographical conditions where it is deployed, as well as frequency band. In this suburban scenario we consider a bandwidth of 2 Mbit/s and range of 700 m. To provide full coverage, it can be shown that the base stations have to be placed in a grid topology, spaced by $\frac{2\times 700}{\sqrt{2}} = 990$ m. All the base stations are connected in a grid topology by ethernet links, and each bus stop is linked to the nearest base station.

There is an Internet server connected to the WiMax base station closest to the center of the grid by 1 Gbit/s Ethernet, while we have 5 mobile terminals attached to each bus and each bus stop. Every time a bus arrives at a bus stop, the 5 terminals inside the bus leave it for the bus stop, while the 5 terminals previously at the bus stop enter the bus. Although 5 terminals per bus and bus stop may seem too few, it allows us to simulate slightly larger networks. However, the analytical model shown further down expands the scope of the main results to a range of terminals per bus and bus stop between 1 and 30 (see Fig. 16).

5.2.2. Analytical model

As in the case of the road scenario, the routing overhead has essentially two components: (1) a variable component that grows with the network size, and (2) a fixed component that is independent of the network size (in case of *bin-dupdate* and explosive solutions). In similar fashion to what was shown in Section 5.1.2 for the "road scenario", as the network grows the variable component also grows proportionally, and the total routing overhead becomes dominated by that parcel. To simplify the analysis, we disregard the fixed component and consider only the variable component. This simplification will be later validated by comparing against simulation results (see Section 5.2.4). In these equations, the $\lceil x \rceil$ notation denotes a *ceiling* operation, i.e. round to the nearest integer larger or equal to x.

As before, we want to find out the amount of MC overhead in the weakest link, i.e. the received bitrate in buses' WiMax links. This overhead is essentially proportional to the total number of Rbridges, and the number of registered terminals in each of those Rbridges. The number of each type of Rbridge is a function of the city area, measured in blocks. To simplify, we consider an area of $h \times h$ blocks.

The bus stops are layed out in a grid, $h/4$ by $h/2$, and so the number of bus stops can be given by:

$$S = \left\lceil \frac{h}{4} \right\rceil \times \left\lceil \frac{h}{2} \right\rceil \tag{20}$$

Along the north and south edges of the grid, each edge with $h+1$ intersections, there are at least $(h+1)/4$ buses. We add the east/west borders, with at least $(h+1)/2$ buses each. Therefore, the total number of buses can be given by:

$$\beta = 2\left\lceil \frac{h+1}{4} \right\rceil + 2\left\lceil \frac{h+1}{2} \right\rceil \tag{21}$$

To compute the number of WiMax base stations we have to consider that they are displayed in a grid and they need to cover the entire city area. The spacing of base stations has to be 990 m, and the city grid has total dimensions given by the Manhattan city block size multiplied by the number of blocks: $h \times 80$ and $h \times 274$. We add one row and one column of base stations to make sure the edges are covered. Hence, the number of base stations is given by:

$$B = \left\lceil 1 + h\frac{80}{990} \right\rceil \times \left\lceil 1 + h\frac{274}{990} \right\rceil \tag{22}$$

The overall number of Rbridges is given by the sum of the expressions (20)–(22). It is easy to see that this sum can be represented as the polynomial expression $C_1 + C_2 h + C_3 h^2$, with C_1, C_2, and C_3 constants. Denoting by A the city area, since $A = h^2$ we may represent the expression as $C_1 + C_2 A^{\frac{1}{2}} + C_3 A$. Considering the asymptotic behavior, we can see that $O\left(C_1 + C_2 A^{\frac{1}{2}} + C_3 A\right)$ simplifies to $O(A)$ under "big O" notation rules. In other words, the total number of Rbridges tends to grow approximately linearly, for large grid sizes.

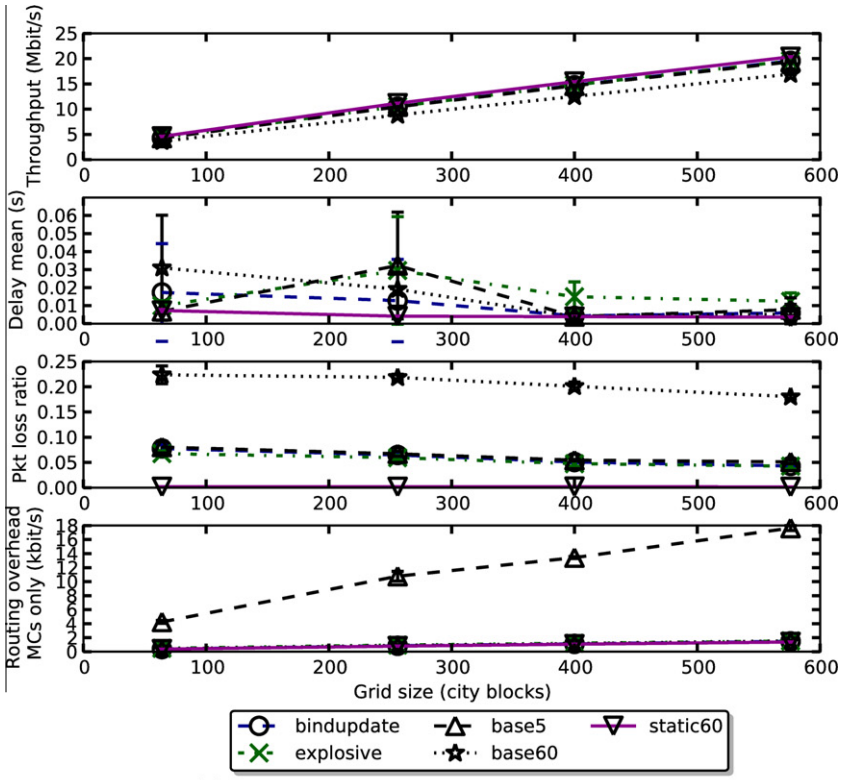
As an example, consider the case of $h = 20$. The grid size would be $20 \times 20 = 400$ city blocks, with an euclidean area equal to $(20 \times 80) \times (20 \times 274) = 8.77\ \text{km}^2$. We would then have 34 buses, 50 bus stops, and 21 base stations. If we would consider to have 5 terminals per bus or bus stop, there would be a total of 420 end-user mobile terminals.

To compute the average number of bus stops connected to each base station, S_B, we multiply the total number of buses by the ratio of base station coverage over total city area, and we obtain Eq. (23). Similar reasoning applies to Eq. (24), which computes the average number of buses in range of each base station, β_B.

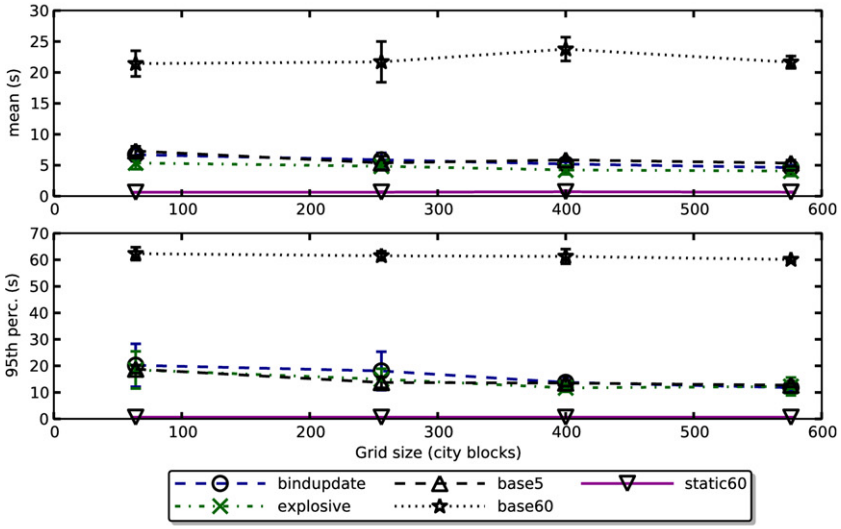
$$S_B = \left\lceil S\frac{990^2}{h \times 80 \times h \times 274} \right\rceil \tag{23}$$

$$\beta_B = \left\lceil \beta\frac{990^2}{h \times 80 \times h \times 274} \right\rceil \tag{24}$$

Finally, we can compute the total overhead produced by Rbridges periodically generating MC messages. Denoting by H the header size of the MC message (20 bytes), and by M the size of each entry inside the MC message (8 bytes), T_S the number of terminals per bus stop (5, in the simulations), and T_β the number of terminals per bus (5, in the simulations):



(a) Throughputs, delays, losses, routing overhead



(b) Flow interruptions

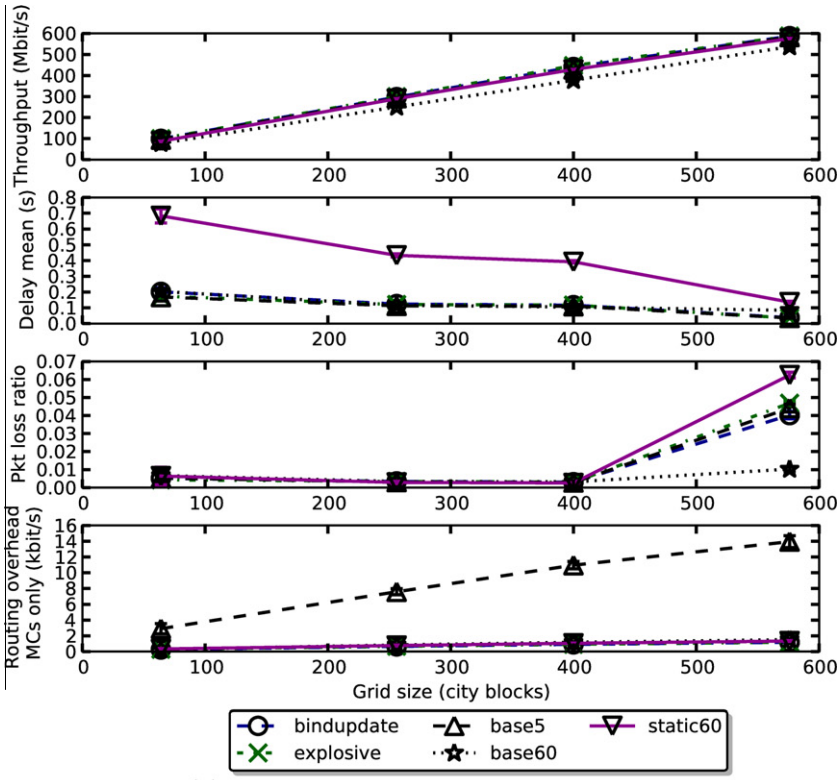
Fig. 13. City grid scenario: UDP results.

$$\text{Overhead} = \mu S(H + (T_S + 1)M) \quad (25)$$
$$+ \mu\beta(H + (T_\beta + 1 + T_B - 1)M) \quad (26)$$
$$+ \mu B(H + (S_B + \beta_B)M) \quad (27)$$

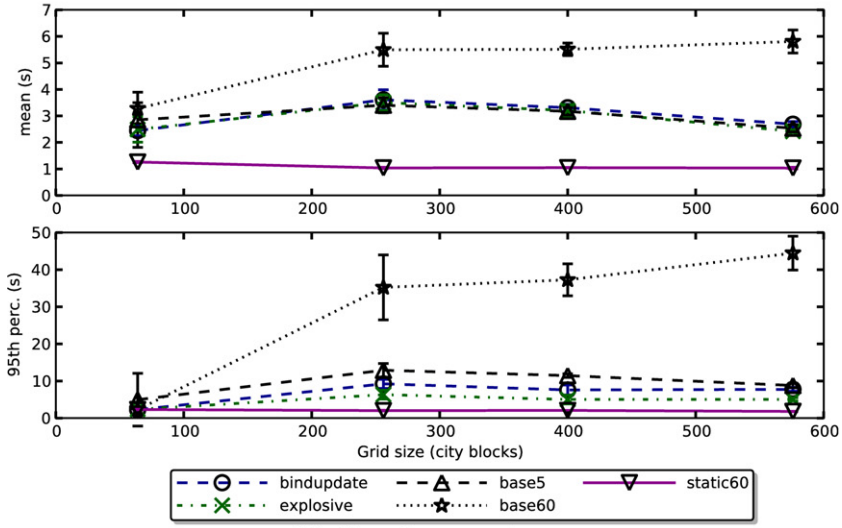
In the above expression, μ represents the rate of MC message generated, i.e. 1/MC interval. The subexpression (25) represents the overhead resulting from MC messages generated by bus stops, (26) the MC overhead generated by buses, and (27) the MC overhead generated by base stations.

5.2.3. Simulation setup

To evaluate the city grid scenario, described in Section 5.2.1, we ran a series of simulations where the grid size is gradually increased. Two sets of simulations were performed. In one set, one 32 kbit/s UDP flow (packet size



(a) Throughputs, delays, losses, routing overhead



(b) Flow interrutpions

Fig. 14. City grid scenario: TCP results.

1000 bytes) was generated by the Internet server, directed to each mobile terminal. In another set of simulations, there is one TCP connection between the Internet server and each mobile terminal, and the server tries to transmit as much data as possible over the TCP connection to the clients. The mobile terminals handover between bus and bus stop, and vice versa, as previously described, and general flow statistics are captured, using the NS-3 Flow Monitor module [27], as well as routing protocol overhead.

5.2.4. Simulation results

Figs. 13 and 14 show some of the simulation results for the city grid scenario with UDP and TCP traffic, respectively. Here we can see that the *base5/bindupdate/explosive*

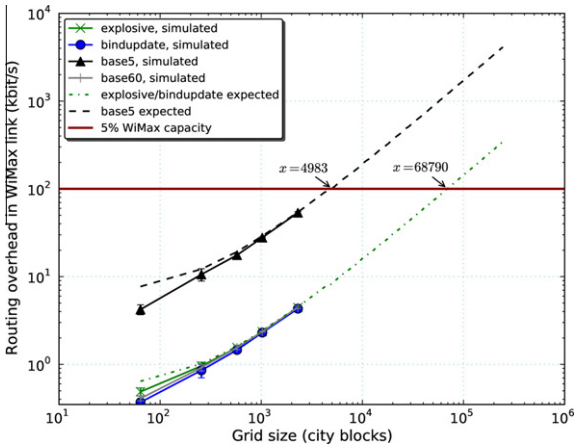


Fig. 15. Grid scenario: routing protocol scalability prediction.

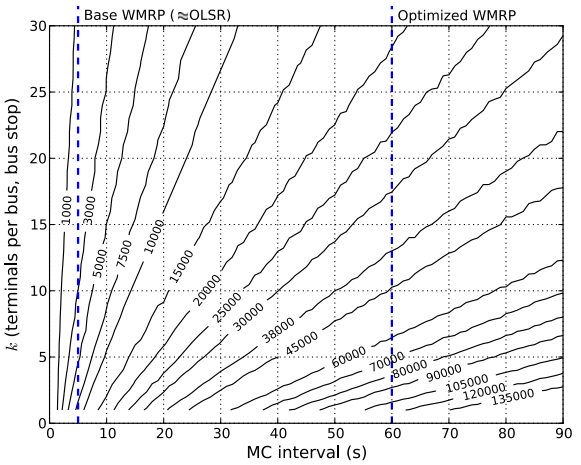


Fig. 16. Maximum grid size (in city blocks) for a range of k and MC interval values.

solutions have nearly indistinguishable performance metrics, whilst the *base60* solution has marginally less throughput and significantly higher packet loss ratio. While the gain of the WMRP optimizations is small for the bitrate/loss/delay metrics, in the flow interruptions metrics the gain is very clear. The *base60* solution has about 22 s of mean interruption time, for the UDP case, while the others (*bindupdate*/*explosive*) have 5 to 6 s mean interruption. In the TCP case, *base60* has about 6 s mean interruption time, while the optimized solutions have about 2.5 s mean interruption. The lower value of mean interruption time in TCP, compared to UDP, may be explained by the apparent flow interruptions that are in fact not cause by handover but by TCP congestion control taking some time to adjust to mobility.

The evolution of the routing protocol overhead with increasing city grid area is shown in Fig. 15. Both analytical and simulation results are represented in the same plot, allowing us to confirm that the analytical model is accurate for large networks. These routing overhead results are obtained for the simulations with UDP traffic, but with TCP traffic the results are identical.

As expected, the *base60* solution generates a small fraction of the overhead of *base5*. Additionally, *explosive* and *bindupdate* have the same control plane overhead as *base60*, but have as good (if not better) user plane behavior as *base5*. Thus we conclude that the optimized solutions have the best of both worlds: good user plane performance, but with a low control plane cost.

5.2.5. Network scalability limit

From Fig. 15 we can find out, for each WMRP configuration, what is the grid area size for which the 5% WiMax capacity limit is crossed. Thus we can see that the non-optimized *base5* solution (similar to OLSR) scales to 4983 city blocks (71×71), which is equivalent to an area of 124 km^2. With the network size at which the threshold is reached, the scenario predicts 630 bus stops, 108 buses, 147 WiMax base stations, and 3690 mobile terminals.

With either of the optimized *explosive*/*bindupdates* solutions, the 5% capacity threshold is crossed at much larger network size of 68790 city blocks (262×262), or 439 km^2. At the maximum network size, we would have 8646 bus stops, 396 buses, 1702 base stations, and 45210 mobile terminals. Compared to the non-optimized solution, the maximum city area increases by a factor of 3.54, and the number of mobile terminals increases by a factor of 12.25.

The obtained limit depends essentially on two parameters: μ and k, considering the simplification that $T_S = T_\beta = k$. Considering that μ is the inverse of the "MC interval", it is easy to determine what is the maximum city grid size that can be attained without surpassing our imposed overhead limit of 100 kbit/s, for a range of MC interval and k values. The obtained scalar field is represented in Fig. 16. As in the road scenario, we can see that the maximum network size increases when the MC interval increases and when k decreases.

5.3. Computational scalability

From the point of view of the control plane overhead *traffic*, it has been shown that WiMetroNet scales well to many thousands of Rbridges and tens of thousands of end user terminals. However, it is not clear how well will a hypothetical implementation handle all the necessary operations in that scale. As possible implementation targets, we are considering two possibilities. The first target is simple DD-WRT wireless router platform, based on *RouterStation Pro Board* containing an Atheros AR7161 MIPS 24 K CPU running at 680 MHz with 128 MB of DDR memory. The second target being considered is based around a more powerful Intel processor, specifically the Intel Atom D510 running at 1.66 GHz.

Regarding the Rbridge data plane, ingress of end user frames is clearly the most difficult operation, specifically the lookup of a destination MAC address in the *Remote Terminal Associations* table. To evaluate MAC address lookup performance, we executed benchmarks on a simple program that does hash table lookups with random MAC-48 addresses as keys. The results in Fig. 17 (bottom) show that

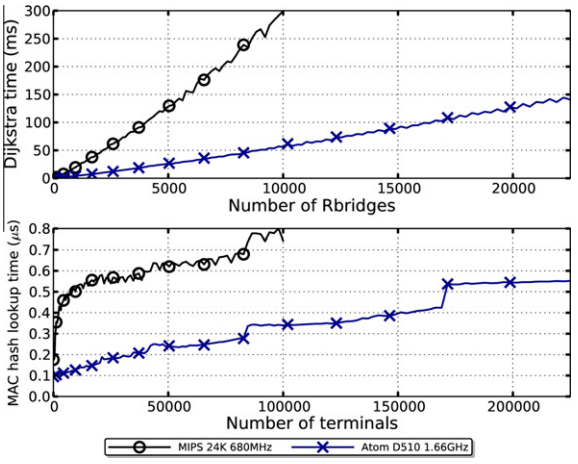


Fig. 17. Benchmark of Dijkstra's algorithm (top), and MAC-48 hash table lookup (bottom).

hash table lookups are below 1μs even for the MIPS CPU and tables with 100,000 entries.

With respect to the Rbridge control plane, the most expensive operation is expected to be the shortest-path computation. In the WMRP routing agent, whenever a TC message arrives, and it represents a change in the topology (rather than e.g. just refresh an existing topology tuple), a flag is set indicating that new shortest paths need to be computed. Once every 250 ms the WMRP agent checks this flag, and if it is set, Dijkstra's shortest path algorithm is run. Thus, in order for the WMRP agent to keep up with topology changes it needs to be able run Dijkstra faster than 250 ms for topologies of 13766 and 10744 Rbridges, which are the network size limits of the road and city grid scenario, respectively. To test this hypothesis, we benchmarked Dijkstra's algorithm on a grid topology while increasing the number of nodes between 100 and 22500 nodes. As shown in Fig. 17 (top), the MIPS CPU reaches the 250 ms limit at approximately 8000 nodes, while the Atom still is very far from reaching it at 22500 nodes. Thus, in an implementation based on the MIPS CPU, shortest path computation becomes a bottleneck that limits the scalability of WiMetroNet, but a router based on the much faster Atom CPU solves this problem. Another alternative would be to make use of dynamic shortest path algorithms. In [28] several dynamic shortest path algorithms are evaluated and concluded to be 10 to 10,000 times faster than repeated application of a static algorithm.

6. Conclusions

In this paper, we have presented a scenario of vehicle networking applied to a public transport system. We show that existing solutions do not work well in this environment, and propose a new architecture that we named WiMetroNet. This architecture entails three separate contributions. First, the user plane that filters broadcasts and optimizes DHCP and ARP traffic via close integration of those protocols with the routing protocol, in contrast with 802.11s and TRILL which do not solve the broadcast issues. Another key difference is that frames are encapsulated using an MPLS header, allowing future protocol extensions without changing the data plane format, and enabling the solution to work also on non-IEEE 802 access links. Second, a new L2.5 routing protocol, that supports both mobile Rbridges and mobile end-user terminals, and feeds the data plane with IP/MAC association tables, much needed for the DHCP/ARP optimizations; a feature that is also missing in 802.11s. Third, routing optimizations to handle fast-handover of terminals in an efficient and scalable way, for large networks. This is something that is not available in either TRILL, 802.11s, AODV, or OLSR.

The new proposals have been demonstrated via simulation and analytical models, and the limits of scalability assessed for two different scenarios: a "road scenario", and a "city grid". For the "road scenario", we have shown that WMRP can scale to at least 2590 bus stops under conservative mobile 802.16 bandwidth estimates, using only 5% of bandwidth for signalling. It is more than enough to cover an entire city's worth of bus lines, which was our initial goal. In the "city grid" scenario, we show that the optimizations increase the covered area by a factor of 3.5 and the number of terminals increases by a factor of 12.25, when compared to a naïve link-state routing protocol, such as OLSR.

As future work, we will be measuring the performance of our optimizations under different types of traffic. We are particularly interested in evaluating how the *bindupdate* solution compares to *explosive* as the number of peers increases, in a peer-to-peer application. We will also be designing and evaluating further optimizations, this time to reduce the routing overhead incurred due to mobility of Rbridges themselves (e.g. handover of a bus from one base station to another one). Further work involves optimizing the network "bootstrap" issues when very long MC/TC/IC refresh intervals are used. Handling temporary disconnection of an Rbridge, which may lead to loss of a periodic global update, will also be improved. We are currently building a prototype Rbridge, to deploy in a small network of ten buses, using a nearly unmodified NS-3 based simulation model and leveraging the NS-3 builtin real-time scheduler and emulation features (EmuNetDevice).

References

[1] M. Ricardo, G. Carneiro, P. Fortuna, F. Abrantes, J. Dias, WiMetroNet, in: 2010 Sixth Advanced International Conference on Telecommunications, IEEE, 2010, pp. 520–525.
[2] A. Myers, E. Ng, H. Zhang, Rethinking the service model: Scaling ethernet to a million nodes, 2004. URL <http://citeseer.ist.psu.edu/myers04rethinking.html>.
[3] T. Clausen, P. Jacquet, Optimized Link State Routing Protocol (OLSR), RFC 3626 (Experimental) 2003, URL <http://www.ietf.org/rfc/rfc3626.txt>.
[4] C. Perkins, E. Belding-Royer, S. Das, Ad hoc on-demand distance vector (AODV) routing, RFC 3561 (Experimental) 2003, URL <http://www.ietf.org/rfc/rfc3561.txt>.
[5] J. Touch, R. Perlman, Transparent interconnection of lots of links (TRILL): problem and applicability statement, RFC 5556 (Informational) 2009, URL <http://www.ietf.org/rfc/rfc5556.txt>.
[6] G. Pei, M. Gerla, X. Hong, LANMAR: landmark routing for large scale wireless ad hoc networks with group mobility, in: Proceedings of the 1st ACM International Symposium on Mobile ad hoc Networking and Computing, 2000, p. 18.

[7] F.J. Ros, P.M. Ruiz, Cluster-based OLSR extensions to reduce control overhead in mobile ad hoc networks, in: Proceedings of the 2007 International Conference on Wireless Communications and Mobile Computing, 2007, p. 207.
[8] IEEE Std 802.11s-2008, March 2008, doi:IEEE P802.11s/D2.0.
[9] Y. Ko, N. Vaidya, Location-aided routing (LAR) in mobile ad hoc networks, Wireless Networks 6 (4) (2000) 321.
[10] K. FENG, C. HSU, T. LU, Velocity-assisted predictive mobility and location-aware routing protocols for mobile ad hoc networks, IEEE Transactions on Vehicular Technology 57 (1) (2008) 448–464.
[11] T. Taleb, E. Sakhaee, A. Jamalipour, K. Hashimoto, N. Kato, Y. Nemoto, A stable routing protocol to support ITS services in VANET networks, IEEE Transactions on Vehicular Technology 56 (6 Part 1) (2007) 3337–3347.
[12] V. Namboodiri, L. Gao, Prediction-based routing for vehicular ad hoc networks, IEEE Transactions on Vehicular Technology 56 (4 Part 2) (2007) 2332–2345.
[13] Y. Fan, J. Zhang, X. Shen, Mobility-aware multi-path forwarding scheme for wireless mesh networks, Wireless Communications and Networking Conference, WCNC 2008, IEEE, 2008, pp. 2337–2342.
[14] A. Capone, S. Napoli, A. Pollastro, Mobimesh: an experimental platform for wireless mesh networks with mobility support, in: Proceedings of ACM QShine 2006 Workshop on Wireless Mesh: Moving Towards Applications, Waterloo (Canada), 2006.
[15] H. Wang, Q. Huang, Y. Xia, Y. Wu, Y. Yuan, A network-based local mobility management scheme for wireless mesh networks, Wireless Communications and Networking Conference, IEEE, 2007, pp. 3792–3797.
[16] Y. Amir, C. Danilov, M. Hilsdale, R. Musaloiu-Elefteri, N. Rivera, Fast handoff for seamless wireless mesh networks, in: Proceedings of the 4th International Conference on Mobile Systems, Applications and Services, ACM, 2006, pp. 83–95.
[17] M. Bezahaf, L. Iannone, S. Fdida, Enhanced mobility management in wireless mesh networks, Journées Doctorales em Informatique et Réseaux (JDIR08), 2008. <http://inl.info.ucl.ac.be/publications/enhanced-mobility-management-wireless-mesh-networks>
[18] L. Couto, J. Barraca, S. Sargento, R. Aguiar, FastM in WMN: a fast mobility support extension for wireless mesh networks, 2009 Second International Conference on Advances in Mesh Networks, IEEE, 2009, pp. 90–96.
[19] L. Lamport, Time, clocks, and the ordering of events in a distributed system, Communications of the ACM 21 (7) (1978) 558–565.
[20] J.P. Macker, J.W. Dean, A study of link state flooding optimizations for scalable wireless networks, Storming Media (2003).
[21] L. Viennot, L. Viennot, P. Hipercom, Complexity results on election of multipoint relays in wireless networks, Internal Report RR-3584, INRIA. URL <http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.43.4536>.
[22] J. HSrri, C. Bonnet, F. Filali, OLSR and MPR: mutual dependences and performances, in: Challenges in Ad Hoc networking, IFIP International Federation for Information Processing, Springer, Boston, 2006, pp. 67–71.
[23] A. Busson, N. Mitton, E. Fleury, Analysis of the multi-point relay selection in olsr and implications, Challenges in Ad Hoc Networking: Fourth Annual Mediterranean Ad Hoc Networking Workshop, June 21–24, 2005, Springer, Île de Porquerolles, France, 2006, p. 387.
[24] D. Kim, H. Cai, M. Na, S. Choi, Performance measurement over mobile WiMAX/IEEE 802.16 e network, in: International Symposium on a World of Wireless, Mobile and Multimedia Networks, WoWMoM 2008, 2008 , pp. 1–8.
[25] Hongqiang Zhai, Younggoo Kwon, Yuguang Fang, Performance analysis of IEEE 802.11 MAC protocols in wireless LANs, Wireless Communications and Mobile Computing 4 (8) (2004) 917–931.
[26] N. Potnis, A. Mahajan, Mobility models for vehicular ad hoc network simulations, in: Proceedings of the 44th Annual Southeast Regional Conference, ACM, 2006, p. 747.
[27] G. Carneiro, P. Fortuna, M. Ricardo, FlowMonitor: a network monitoring framework for the network simulator 3 (NS-3), in: Proceedings of the Fourth International ICST Conference on Performance Evaluation Methodologies and Tools, ICST, 2009, pp. 1–10.
[28] C. Demetrescu, G. Italiano, Experimental analysis of dynamic all pairs shortest path algorithms, ACM Transactions on Algorithms (TALG) 2 (4) (2006) 578–601.

Gustavo Carneiro (gjc@inescporto.pt) received a Diploma degree in electrical and computer engineering from Porto University, Portugal, in 2001 and since then has been a researcher at INESC Porto. He actively participated in the European IST ARROWS, and IST DAIDALOS (1 and 2) projects and currently is participating in the European ALICANTE project. He received an MSc diploma in 2006, and is actively pursuing a Ph.D. at Porto University, Portugal.

Pedro Fortuna received a Licenciatura degree in Computer Science in 2002 from the Polytechnic Institute of Porto, School of Engineering (ISEP), Portugal. In 2007, he received a M.Sc in Computer Networks from the University of Porto, Portugal. He works as a researcher at INESC Porto and since 2006 he is pursuing his Ph.D. in Electrical and Computer Engineering. His research interests include Network Mobility, Ad-Hoc Routing Protocols, and IP Header Compression. Currently he is an assistant professor at ISEP.

Jaime S. Dias (jdias@inescporto.pt) received his licentiate degree in electronics and computer engineering from the Polytechnic Institute of Porto, School of Engineering (ISEP), Portugal, in 1998. He received his MSc degree in networks and communication services from the Engineering Faculty of the Porto University, Portugal, in 2005. He is currently a researcher at the INESC Porto and pursuing his Ph.D. degree in electrical and computer engineering at the Engineering Faculty of the Porto University, Portugal. His current research interests include system and network security, future internet, and mobile networks.

Manuel Ricardo received a Licenciatura degree in 1988, a M.Sc in 1992, and a Ph.D. in 2000, all in Electrical and Computer Engineering from the University of Porto, Portugal. He is currently an associate professor at the University of Porto, where he gives courses on wireless and computer networks. He also leads the Wireless Networks research area of the INESC Porto research institute.