A cross-layer protocol for cooperative content delivery over mobile ad-hoc networks

Claudio Greco and Marco Cagnazzo*

Institut TELECOM, TELECOM-ParisTech, TSI department
46 rue Barrault, F-75634 Paris Cedex 13, FRANCE
E-mail: {greco,cagnazzo}@telecom-paristech.fr
*Corresponding author

Abstract Real-time multimedia streaming over MANETs is an active research field since this technology promises scalable and robust audio/video delivery without infrastructure. Even though this problem has several commune features with peer-to-peer routing, the additional sensible parameters of MANETs make wired solutions unfit to this case; therefore, we propose a content routing/delivery protocol inherently designed for the ad-hoc wireless case, exploiting the intrinsic broadcast property of the medium. We provided an implementation of this protocol and we tested it in several use-cases, observing how it assures availability, robustness, and scalability.

Keywords: Ad-hoc networks; MANETs; Peer-to-peer; Cross Layer; Multiple Description Coding; Neighbour Knowledge; Cooperative networks.

Biographical notes: Claudio Greco obtained his MSc in Computing Engineering in 2007 from "Federico II" university (Naples, Italy). He is currently a PhD candidate at TELECOM-ParisTech (Paris, France). His research interests are multiple description coding, cooperative networking, video streaming.

Marco Cagnazzo obtained his PhD in Information and Communication Technology from the "Federico II" university (Naples, Italy) and the university of Nice-Sophia Antipolis (France) in 2005. He is currently maître de conférences (roughly equivalent to associate professor) at TELECOM-ParisTech. His research interests are multiple description coding, distributed video coding, multiview video coding.

1 Introduction

A mobile ad-hoc network, or MANET, is a dynamic network of self-organising mobile devices, connected by wireless links in a mesh topology, with no pre-existing infrastructure [Frodigh et al., 2000]. MANETs present several desirable properties that qualify them as an attractive topic for the research community: *flexibility*, *ease of deployment*, and *robustness*. The MANET model – equally, poorly equipped nodes that self-organise their topology in order to locate and route a piece of information – shares obvious similarities with peer-to-peer (P2P) network model. For instance, the two models have similar routing principles (such as using broadcast

and flooding), similar topologies (i.e., flat, changing topologies with a high churn rate), and a low reliability of single nodes.

Nevertheless, there are several important parameters of ad-hoc networks that are usually not taken into account by wired peer-to-peer protocols, such as node mobility, link quality, and node density. On the Internet, a P2P network is an overlay justified by the need for specialised functions not possible at IP level, e.g., multicast support (not provided by most IP routers). Unstructured P2P overlays offer a logic network through which queries are flooded when a peer looks for piece of data to find as many peers as possible sharing the data. Structured P2P overlays offer a content addressable networks that permit efficient content-based routing, which is otherwise not possible in the IP layer.

However, MANETs are based on a network architecture that is quite different from the Internet architecture [da Hora et al., 2009]. The Internet architecture consists of active nodes at the edge of the network exchanging messages through an infrastructure of passive routers in the core. Routing is based on addresses in a hierarchical way. A typical MANET architecture resembles more to the architectures introduced in the 80s to interconnect LANs, with routing based on MAC addresses. The MANET routing functionality is non-hierarchical on the MAC address and is several layers below the functionality offered by a P2P network. This suggests that in a MANET the P2P concept should be implemented on top of MAC level routing. Dropping the traditional layered approach typical of network protocols to move toward a cross-layer design is the current trend in P2P-over-MANET design [Setton et al., 2005]. Cross-layer design allows to enhance the end-to-end performance of the protocol passing key parameters across the protocol stack, at the price of an increased complexity.

Since MANETs are usually built to support a specific application (application-driven networking), in our scenario we may safely assume that all nodes are interested to whatever stream is distributed and willing to cooperate to its distribution; hence, there is no need to flood any request: when a stream is distributed, the goal of the source is to reach any other node of the MANET as quickly and as reliably as possible. Unstructured P2P protocols over the Internet spread their messages by flooding them on the overlay network, but on MANETs the overlay should be the network itself: logical connections must match the MAC layer connections.

Our contribution consists in a P2P cross-layer protocol able to deliver video content through a mobile ad-hoc network. The protocol is meant to provide an application-driven MAC-level selective flooding that can efficiently relay a real-time video stream represented in multiple descriptions.

The literature provides a fairly wide number of articles in the context of multimedia streaming over mobile ad-hoc networks, which has been a lively topic of research for some years now. Setton, et al. [Setton et al., 2005] provided a cross-layer design framework in order to integrate congestion-distortion scheduling into ad-hoc networks, a technique they extensively studied in the past [Setton, 2007]. Da Hora, et al. [da Hora et al., 2009] investigated on how peer-to-peer content delivery can be optimised for mobile ad-hoc networks. Their work proved how unstructured protocols offer a higher resiliency at the price of a loss in scalability, a worst power consumption, and an increased end-to-end delay, if compared to structured protocols, which are more energy-effective, but show a degradation in performance in more dynamic networks. Williams and Camp, in their survey of

broadcasting techniques for ad-hoc networks [Williams and Camp, 2002], categorise flooding algorithms in four families: Simple, Probability-Based, Area-Based, and Neighbour Knowledge-Based; our work takes its place in the latter family. All these works provided some very interesting points; nevertheless, none of them appears as a prefect match to our specifications. This work is somehow related to the work of Kunz [Kunz, 2003]; the key difference being that in our scenario only one source for the video stream is considered, so the mesh structure of the overlay can be simplified in a tree structure.

As far as the problem of real-time transmission over unreliable network is concerned, a very popular solution is multiple description coding (MDC) [Goyal, 2001]. MDC is a framework allowing a partial immunity to the inevitable loss of packets on unreliable networks, in particular on peer-to-peer wireless networks. When using MDC, one tries to trade-off coding efficiency (in terms of compression ratio for a given quality) with robustness. There exist several ways to achieve MDC, based on the use of unconnected quantization cells [Vaishampayan, 1993], on correlating transforms [Wang et al., 1997] and on redundant transforms [Goyal et al., 1999]. A wavelet-based video coder both progressive and MD coding have been introduced [Tillier et al., 2007]. Further protection from errors could be achieved by using network coding [Chou et al., 2003].

The proposed protocol is independent from the actual MD coder, and in the following we shall refer to a generic MDC scheme producing a certain number of descriptions. Each user can improve the quality of the reconstructed signal by increasing the number of received descriptions. Moreover, we shall show that our protocol provides a certain degree of diversity for the paths associated to different descriptions, improving the robustness of the transmission.

The rest of this article is organised as follows: in section 2, we describe our protocol's structure and the adjustment to the MAC layer it may require; in section 3, we see in detail the evolution of the protocol; we show in section 4 an experimental study of the protocol; section 5, draws conclusions and plans future work.

2 ABCD Protocol

We want to introduce a fully operational cross-layer protocol, capable to offer complete coverage of an ad-hoc wireless network of mobile devices with a real-time video stream. We called this protocol ABCD: A Broadcast Content Delivery Protocol.

Our protocol must jointly select both server and router, an approach that showed a significant gain in video quality [Mao et al., 2007]. To the nodes' benefit, we aim to increase the quality of the received video stream by maximising the number of received descriptions and the signal-to-noise ratio on the last link, while minimising the number of hops between each node and its source of the stream. However, to the network's benefit, we also aim to minimise the number of messages sent for streaming and set-up: we try to minimise the number of nodes transmitting the stream, and to maximise the information a node can retrieve without making any request. Let us focus on this latter point: nodes should be able to gather information without making explicit requests. One obvious solution to this problem is that they should be able to read other nodes' messages and extract information from there.

The wireless medium is inherently broadcast, so each node is normally receiving (more properly speaking, *sensing*) a certain number of packets for which it is neither the sender nor the designated receiver. Let us define *neighbours* of a node n all the nodes whose sent packets can be sensed by n. The neighbourhood of a node can change in time because of mobility and churn. However, sensing a packet and reading the encapsulated message are two different matters. The 802.11 standard MAC layer was not designed to provide an *open* channel, so it provides instead mechanisms to force point-to-point communication over a broadcast medium. To achieve full disclosure among neighbours, we need get around the 802.11 standard and provide a modified MAC layer more suited to our purposes.

2.1 MAC Layer Modifications for ABCD

The goal of our modified MAC layer is to make the nodes able to read messages that were not meant to them. Let us assume that the MAC layer is able to identify packets that encapsulate ABCD messages, which is legitimate since we are operating under the hypotheses of a cross-layer protocol. Let us also assume that non-ABCD packets are dispatched to the standard 802.11 MAC layer.

For ABCD packets, we set the destination fields in both the 802.11 frame and the IP datagram to their respective broadcast addresses, while the *actual* recipient is specified in the application level message with a unique ID (the problem of assigning and resolving unique application level IDs is outside has known solutions [Waldvogel and Rinaldi, 2003]). This technique obviously force any ABCD message to be received (thus possibly read) by any neighbour of the sender; however, it raises an ulterior problem: as mentioned early, 802.11 was mainly designed for one-to-one communication, and one-to-many communication is known to be unreliable and inefficient [Tourrilhes, 1998]. Whereas this is normally not considered to be a major concern for other protocols, we want to make an extensive use of broadcast, so a form of broadcast reservation, for simple it may be, is in order. A very straightforward – yet effective – solution can be found in the paper by Marina et al. [Marina et al., 2001]: it consists in forcing an RTS/CTS/ACK scheme for broadcast packages also. The choice of the node with whom the RTS/CTS/ACK exchange is performed is discussed in section 2.4.

2.2 Protocol States

The open channel provided by the modified MAC layer is meant to make the nodes able to gather information about the availability of the resource in their *neighbourhood*, i.e., the set of their neighbours. This information needs to be represented in a compact form, to be easily accessible for a real-time decision support. For each description, we introduce four states; at any time each node will be in one of these four states, and its state will depend on its knowledge about the description.
State A: the node is currently receiving and re-transmitting the description to one or more nodes;
State B: the node is currently receiving the description but it is not re-transmitting it. This implies that it has at least one neighbour in state A;
State C: the node is currently aware of at least one node receiving the description. This implies that it has at least one neighbour in state B;
State D: the node has no knowledge of the description.

Any node will keep, for each description, a list of its neighbour and their states (details in section 3). Any time a node inspects a message from one of its neighbours, it can infer its state and its satellite information (see section 2.3); the corresponding tables do not report which state the sender actually is in, but which state can be inferred with a pessimistic look. However, until a new message arrives, the node cannot know whether the information it has is still consistent. To handle the expiration of information, we proceed as follows. Let us assume node n_0 receives a message m, sent by node n_1 (whether the message was sent to n_0 or not is irrelevant). A state s is associated to m according to table 1. If node n_0 does not have n_1 in its list of neighbours, it adds the couple (n_1, s) to the list and starts a timer $t(s)$, whose time interval depends on the state s. When the timer expires the information carried by message m about state s is considered expired; nevertheless, node n_1 is not necessarily removed form the list: instead, its state is updated to state $s' = s + 1$ and a corresponding timer $t(s')$ is started (we define $B = A + 1$; $C = B + 1$; and $D = C + 1$). If and only if the current state is D when the timer expires, node n_1 is removed from the list of neighbours. On the contrary, if n_0 does have n_1 in its neighbour list, it just starts (or re-starts) the associated $t(s)$ timer. We call this *rejuvenation* of state s.

2.3 ABCD Protocol Messages Overview

As mentioned above, all the messages of the protocol are sent with Data Link and Network Level destination addresses set to broadcast; however, at Application Level, we need to tell which messages are intended to whom, even if its content will be readable by any node in the sender's neighbourhood.

In order to do so, we classify the protocol messages in three categories: Unicast, Multicast, and Broadcast messages (see Tab. 1).

Message	Sender's State	Destination	Semantic
Advert	A	Broadcast	Used to advertise a stream
Data	A	Multicast	Contains video packets
Attach	B	Unicast	Subscribe to a group
Detach	D	Unicast	Unsubscribe from a group
Leave	D	Multicast	Disband a group

Table 1 Protocol Messages

Unicast messages are supposed to be received by at least one node, whose ID is specified in the Application Level address field. Reception by any other node is collateral, and does not affect the proper evolution of the protocol (however, it does affect the *efficiency* of the protocol, as we shall see later). In this category, we find Attachment and Detach messages.

Multicast messages are supposed to be received by a group of nodes. However, this group is known only to the sender, and is not specified in the message. In the Application Level address fields, there is specified the ID of one node only, which we define as the *group leader* for the message. The message is considered properly transmitted if and only if the group leader receives it properly. The protocol is able to handle the loss of the message by other members of the group. In this category, we find Data and Leave messages.

Broadcast messages are sent with no *a priori* knowledge on whom shall receive the message. In the address fields, there may be specified either a node's ID or the special ID *PEER_NULL*. If a valid ID is specified, we define the corresponding node *control peer*. A control peer for a broadcast message is basically like a group leader for multicast message. In this category, we find Source messages.

As we shall see in section 2.4, these categories are used to determine the behaviour of broadcast reservation at MAC Level.

2.4 Broadcast Reservation vs Message Category

As anticipated in section 2.1, we want to force RTS/CTS/ACK exchange on broadcast packets, but the question remains on whom the exchange is performed with. For each message, the RTS/CTS/ACK exchange is performed with the peer whose ID is specified in the Application level address field, how this peer is selected depends on the encapsulated message category.

For unicast messages we just ignore that the packet will be sent in broadcast, and perform the RTS/CTS/ACK exchange with the designated recipient.

For multicast messages, the RTS/CTS/ACK exchange is performed with the group leader. Various strategies are possible in selecting the group leader, the only constraint being that the leader must actually belong to the group. A simple strategy, which we implemented, is to choose the node in the group with the best signal-to-noise ratio to the sender (as measured at the sender).

Broadcast messages may or may not specify a control peer. The control peer is chosen with the same strategy as the group leader, but on the entire set of neighbours of the sender. Then, it can be used to perform the RTS/CTS/ACK exchange. However, if the neighbour set is empty, the PEER_NULL special address is specified; in this case, the sender cannot perform any RTS/CTS/ACK exchange, thus cannot reserve the channel for broadcast. Instead of reserving the channel, the sender may try to make the transmission more robust by sending the message multiple time (to maximise the odds that the message shall be eventually received by some node). Under the hypothesis that the traffic generated by ABCD is the only traffic in the network (consistent, for instance, with a disaster or war scenario) an empty neighbourhood implies silent neighbours, thus it is very unlikely that the broadcast repetition could cause any collisions.

3 ABCD: A Use Case

We shall here describe how a new node joins the overlay network. Let us assume we have a running instance of the ABCD protocol deployed on a MANET. For simplicity's sake, let us assume also that the stream is coded with only two descriptions, that we shall call *red* and *blue*.

Before starting the streaming, the source generates and sends repeatedly an Advert message to inform its neighbours that a stream will shortly be available. This is done in order to grant the nodes the possibility of building the delivery trees before actually starting the stream, thus preventing the loss of the first packets of the stream. From the protocol's point of view, an Advert message is exactly like a Data message (it implies the same state and conveys the same satellite information), except that it does not contain actual video data.

Upon connection, each node chooses which description it will try to get. The choice of the next description to get shall be explained in section 3.3. Let us assume that the node choices to get the red description; at this stage the node does nothing but set a timer, and start a *listening* period.

In this period, the node will listen to the channel, inspecting all the messages concerning the red description, filling the node's lists with useful information about its neighbours.

As the listening interval expires, the node examines its list of neighbour on the red description. Several cases are possible at this stage:
1) The node has at least one A peer among its neighbours;
2) The node has at least one B peer among its neighbours, and no A peers;
3) The node has neither A nor B peers among its neighbours.

In case 1, the node is within the range of transmission of a node that is already multicasting the red description: it can simply listen the channel, and automatically receive it. However, to prevent the sender to stop the multicast, the node has to notify it with its intention to receive the description, subscribing to its multicast group. If there is more than one peer in state A in its neighbourhood, it will subscribe to only one of them (peer selection strategy is discussed in section 3.2).

Subscription to a multicast group is performed by sending periodical Attach messages to the multicasting peer. In lack of such a message, the multicasting peer will remove the node from the multicast group, and could eventually interrupt the transmission if the multicast group gets empty; therefore, in order to minimise the number of active nodes, our protocol encourages peers to attach to nodes having large multicast groups; also, the Attach messages have the side-effect of advertising node's state (in this case, B) to its neighbours.

In case 2, the node is aware of at least one neighbour receiving the description (e.g., it intercepted an Attach message). To start receiving the description, the node has to request its peer to start a multicast group. This is also done via an Attach message, which will also advertise its state (B) to its neighbours. Again, even if more neighbours are in B state, the Attach message will be sent to only one of them (see section 3.2). If the requested peer does not start to send the description, the node will send an Attach message to another B peer if such a peer exists. If no peer starts sending, we go to case 3.

In case 3, the node has been unable to determine whether any of its neighbours is in state A or B, so the node enters a sleep state for a random time, then retries to connect as described through this section.

3.1 Dynamic Parent Switch

It may happen at some point that a node has two or more neighbours relaying the same description in its neighbourhood, perhaps solicited from two different peers or because of nodes' mobility. The node is able to realise this because, thanks to the open channel, it reads the Data messages of both multicast groups (see 2.1).

If this happens, it may decide to abandon its current multicast group to subscribe a better one. We call this procedure *parent switch*. It occurs when the new candidate parent has a smaller number of hops from the source or, the same distance from the source and a larger multicast group.

Switching parent allows continual improvements in topology: as soon as a better topology is made available (by mobility or churn) and known (by periodic attachment messages) it is achieved.

3.2 Best Peer Selection

In several phases through the execution of ABCD, a node might have to choice among different peers, e.g., to demand to start a multicast group, or to join an existing group. Peer selection is performed using a total order relation over the set of neighbours. This relation is a lexicographical order relation defined on pairs of nodes (n_0, n_1) that allows to sort the list of neighbours. The first in the sorted list will be chosen as the *best* peer.

The order relation can be defined with the following algorithm:
1) Choose the node with higher state (A, B, C, D);
2) Choose the node with the smallest number of hops from the source;
3) Choose the node node with the largest multicast group;
4) Choose the node node with the highest signal-to-noise ratio;
5) Choose the node node with the smallest ID.

Where each rule is applied as a tie-break for the previous. Note that, since IDs are unique, the order relation is complete.

3.3 Best Description Selection

As mentioned above, a node occasionally has to choose which description try to get. In order to minimise connection delay, the node inspects its lists of neighbours for each description; the availability of each description is estimated with the state of the best peer (as defined in section 3.2) in each list.

Notice that the nodes will always try to get all the descriptions, just in a different order. Description selection allows to join the overlay as soon as information about one of the multicast trees is available; in the meanwhile, the node is able to gather information on the others trees. This technique allows to reduce the frequency of protocol messages, thus reducing the protocol overhead.

3.4 Node Disconnection

When a node decides to disconnect from the overlay, it takes some precautions to minimise the bother caused to its neighbours. Namely, it explicitly unsubscribe the multicast group (with a Detach message) for the descriptions it is receiving, and disbands the multicast groups (with a Leave message) for the descriptions it is relaying, if any. However, the protocol can manage abrupt disconnection and, in the same way, departures due to mobility, inferring detachment or leave of a node by its lack of periodic Attachment or Data messages.

3.5 Overhead Control

To keep nodes subscribed to a group, some of the attachments could actually be useless, since active nodes broadcast the description regardless the number of subscribed nodes, provided that it is greater than zero. Thus, we let nodes send their

Propagation Model	TwoRayGround
Carrier	2.472 GHz (Channel 13)
Transmitted Signal Power	15 dBm
Collision Threshold	10 dB
Receiver Sensitivity	−82 dBm
Nominal Range	25 m

Table 2 Simulated Network Adapter Specifications

attachment messages with a given probability decreasing with the number of nodes subscribed to the active node, which is notified in the Data packet header.

Also, in order to reduce congestion on the network, we decrease the frequency of attachment messages as nodes stay subscribed to the same active node. The reason for that lies in the fact that when a node changes its active node (or *parent*, nodes around it may also benefit from a change in their subscriptions, to adapt to the new environment. However, after a short while, the overlay network will reach a stable topology and attachment messages can become rarer to reduce network congestion.

4 Experimental Study

In order to validate and test our protocol, we provided an implementation for the discrete event simulator *ns2* [ns2], which accurately models the 802.11 MAC/PHY with collisions, hidden nodes, interference, etc.

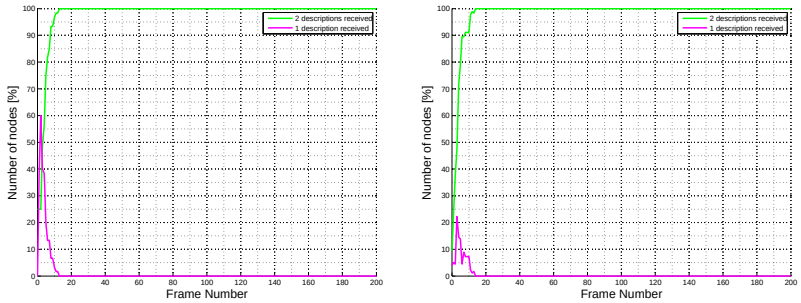
Our implementation consists in a modified version of 802.11x MAC layer agent to support robust broadcast, and a routing agent which implements the application logic. Mobile nodes parameters are based on the specification of Orinoco 11b Card [ori, 2004] (See Table 2). The stream to be broadcasted is a video sequence encoded in multiple descriptions (two descriptions) of 200 frames per description, with total average bitrate of 384 kbps (consistent with videoconferencing quality).

Several sets of tests have been performed, with number of nodes varying from 10 to 200. For brevity, we report here only a few significant scenarios, with reference number of nodes of 100.

In the first two scenarios, nodes are randomly located with uniform distribution with a density of approximately 6.5 nodes per squared transmission range, which is close to the ideal value that maximises the normalised network throughput [Kleinrock and Silvester, 1978]. Results are shown in Figure 1 for two network with same node density and different sizes. It should be noticed that in both cases 100% of nodes is receiving at least one description by frame 10 and both of them by frame 15 (i.e., after 600 ms and 1 s respectively at 15 fps). The maximum delay between the source and a node is in the order of 150 ms.

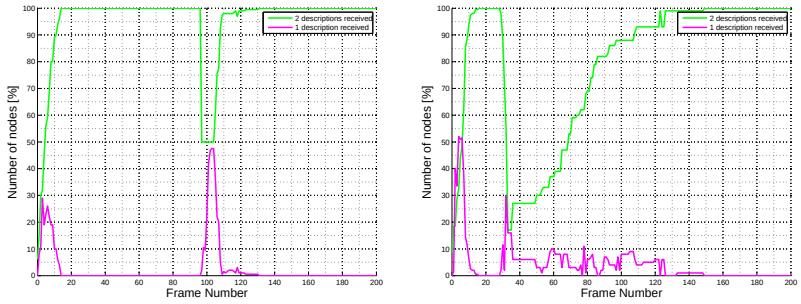
In the third scenario, 100 nodes connect instantaneously as the 100-th frame is sent (Figure 2-(a)). After the number of nodes doubles, 100% of nodes is receiving at least one description in 15 frames and both of them in 30 (i.e., in 1 and 2 seconds respectively). The system deals very well with a conspicuous number of nodes connecting, as a result of the broadcast channel once the area has been covered, the marginal effort to attach the new nodes is small.

Conversely, in the fourth scenario, 100 out of 200 nodes disconnect as the 33-rd frame is sent (Figure 2-(b)). Even though the abrupt disconnection is a very unlikely



(a) (b)

Figure 1 (a) Received frames for 60 nodes in $76 \times 76\ m^2$. (b) Received frames for 180 nodes in $132 \times 132\ m^2$.



(a) (b)

Figure 2 (a) Received frames for 100 nodes in $140 \times 140\ m^2$; 100 more nodes connect on the 100-th frame. (b) Received frames for 200 nodes in $140 \times 140\ m^2$; 100 nodes disconnect on the 33-rd frame.

scenario indeed, we use this to test the robustness of the protocol. Here, 100% of remaining nodes is receiving at least one description in about 90 frames and both of them in about 95 (i.e., in 6 and 6.25 seconds respectively). This relatively long time is due to the necessity for the remaining nodes to flush their current neighbour lists (invalidated by the disconnections) and fill them in with valid information. Strategies to speed-up reconnection after a catastrophic event are pointed-out in Section 5.

The number of active nodes needed to achieve full coverage depends on both number of nodes and topology. Average values for different numbers of nodes are presented in Table 3.

In Figure 3, we show the topology of the overlay network for a scenario of 80 nodes in $88 \times 88\ m^2$. The path diversity between descriptions should be noted. Also, it is worth mentioning that the protocol achieves this diversity with no explicit mechanism to enforce it, but relying on the probabilistic send of attach messages. Strategies to enforce diversity are outlined in Section 5.

The end-to-end delay properties of the protocol are worth mentioning; Figure 4 shows maximum and average end-to-end delay of two different scenarios. In Figure 4-(a), we have 60 nodes that connect the overlay uniformly in $t \in]0, 1]$ and stay

Number of Nodes	10	20	40	60	80	100
Active Nodes (at least one description)	1.0	4.5	10.5	19.5	27.5	33.0
Active Nodes (both descriptions)	1.0	2.5	7.0	10.5	14.5	19.5

Table 3 Average Number of Active Nodes

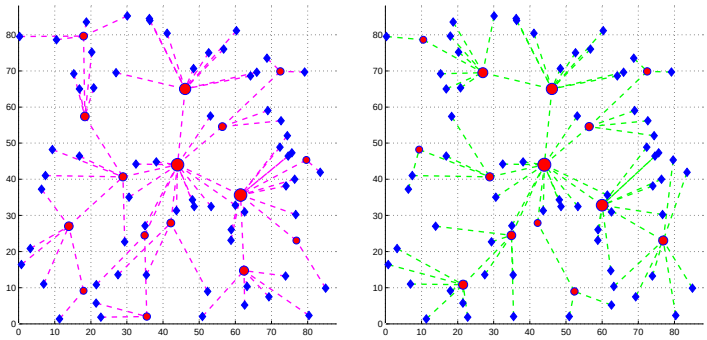


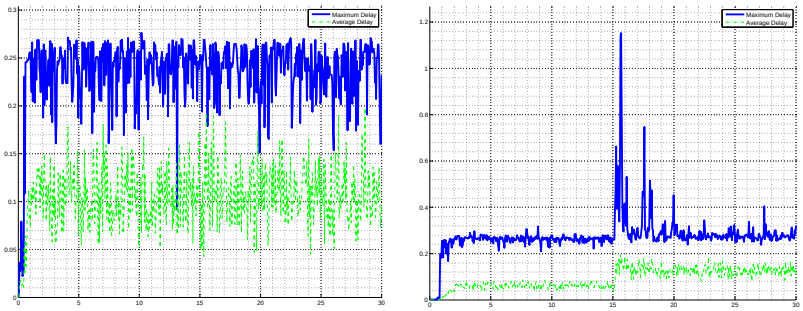
Figure 3 Diffusion trees for 80 nodes in 88×88 m^2, two descriptions. Round spots represent active nodes, the size of the spots is proportional to the number of nodes subscribed.

connected the whole duration of the simulation; this scenario is used to represent the regime behaviour of the protocol when churn rate and mobility are small. We notice that average and maximum delay are about 100 and 250 milliseconds respectively, which are acceptable delays for live streaming. In Figure 4-(b), we have 100 nodes that connect the overlay uniformly in $t \in]0, 1]$ and 100 more that connect in $t \in]15, 16]$; they all stay connected untile the end of the simulation. In this case, we notice two things: first, the average delay increases when the number of nodes doubles, but still stays lower than 160 milliseconds; the maximum delay stays more or less equal at 250 milliseconds with both 100 and 200 nodes, however it spikes to more than 1 second while they are in the process of connecting. This is due to the fact that the channel is being contested between video packets (Data) and topology packets (Attach). Even though this congestion leads to some packet to be dropped due to excessive delay, multiple description coding is able to mask this transitory effect during the time needed to the congestion to fade out.

As mentioned in section 3.5, we try to keep the packet overhead low by increasing the topology messages exchange when topology changes occur, then slowly reducing it overtime. The drawback of this technique, is that when massive changes occur in a short time, the load of the network grows rapidly, causing congestion quickly, but for a very short time (compare with Fig. 4-(b)).

In Table 4, we measure the packet overhead for a network of 140×140 m^2 with various number of nodes. To force a massive topology change, the nodes doubles at half of the simulation time. We define the packet overhead of the protocol as (Number of Data Packets + Number of Attach Packets)/(Number of Data Packets).

The reason we choose to measure packet overhead instead of byte overhead, is that in our scenarios the former is more correlated with end-to-end delay, since the time to gain access to the channel is the bottleneck of the system.



(a) (b)

Figure 4 End-to-end delay versus simulation time (both in seconds). (a) Scenario with 60 nodes in 76×76 m^2 (compare with figure 1-(a)). (b) Scenario with 100 nodes in 140×140 m^2; 100 more nodes connect on $t = 15$s (compare with figure 2-(a)).

Number of Nodes	20	30	40	50	60
Packet Overhead	2.02%	2.89%	4.62%	5.73%	5.91%
Number of Nodes	70	80	90	100	110
Packet Overhead	6.42%	6.94%	8.70%	9.16%	9.62%

Table 4 Packet overhead for various numbers of nodes in 140×140 m^2, number of nodes doubles at half of the simulation time.

Finally, in figure 5, we show a comparison of the delivery rates of our protocol with two different flooding protocols: Simple Flooding and Probabilistic Flooding (see also [Williams and Camp, 2002]).

Consistently with our expectations, Simple Flooding achieves full coverage of the network, but the actual delivery rate is affected by the numerous collisions due to the high overhead. Probabilistic Flooding, on the other hand, reduces the collision, but fails to cover the entire network (notice that different trade-offs are achieved with different values of p, but the general principle holds). ABCD, being a Neighbour Knowledge-Based algorithm, is able to reduce the overhead without affecting the coverage, and to dynamically adapt to the local topology.

All tests presented here have been performed with stationary nodes, this model permitting to focus on a number of performance indicators (coverage, protocol overhead, delay, etc.) reducing the number of axes to be explored in the parameter space (mobility model, speed, etc.). Nevertheless, stationarity (or quasi-stationarity) is a realistic model only in some of the possible scenarios of application. We have performed a preliminary set of tests using a generalized random waypoint model [Palchaudhuri et al., 2005] with average speed up to *ca.* 10 meters per second that shows no relevant change in performance. However, mobility is a topic which requires an extensive study, incompatible with the space constraints of this article, and has therefore been left out of its scope, to be investigated in future works.

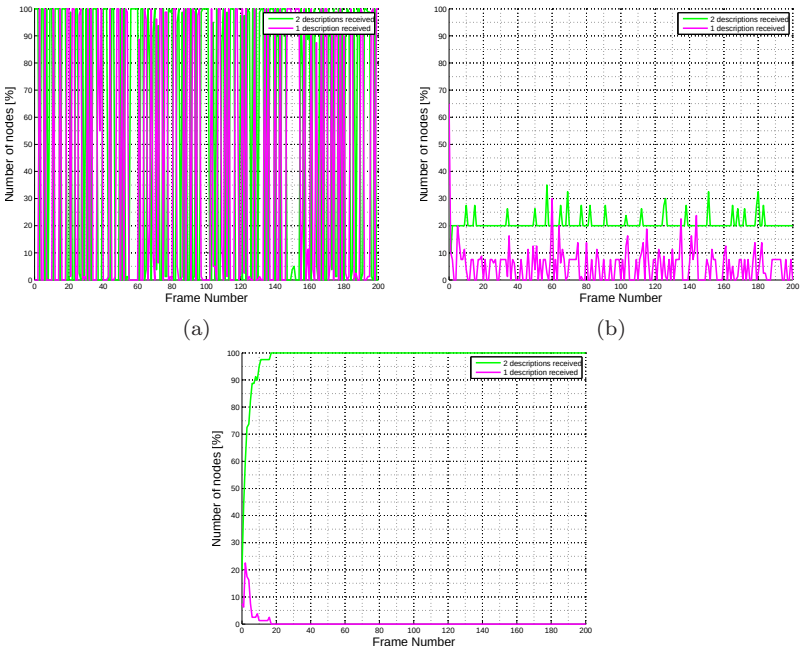


Figure 5 Comparison of delivery rates. (a) Simple Flooding. (b) Probabilistic Flooding with $p = 20\%$. (c) ABCD Protocol.

5 Conclusions and Future Work

Mobile ad-hoc networks have been an attractive topic for the research community for some years, due to their interesting properties of flexibility, ease of deployment, robustness, and heterogeneity. The MANET model has been often likened to the peer-to-peer model, with which it shares similarities in terms of routing principles, topology, and a low reliability of nodes; however the P2P does not take into accounts parameters as node mobility, link quality, and node density, which are crucial for ad-hoc wireless scenarios.

In this paper, we provided a peer-to-peer protocol designed to fill this gap and provide an overlay network that efficiently relays a real-time video stream when deployed on a mobile ad-hoc network. To assure diversity and robustness, we used Multiple Description Coding to split the stream and routed each sub-stream separately. Our results showed how the protocol is efficient, robust, and scalable providing that certain conditions on node density are met. In particular, we saw how ABCD performs better on highly dense networks, where it can benefit the most of the inherent broadcast nature of the medium.

As future work, we shall study the integration of ABCD with the specific video codec used for the stream, in order to optimise them jointly. In particular, we shall focus on Wavelet Based Multiple Description Coding [Tillier et al., 2007]. Also, we shall try to introduce video quality metrics into ABCD, i.e., bias the random walk search with parameters depending on the video quality, such as the video distortion.

References

The Network Simulator – ns-2. Website. URL `http://www.isi.edu/nsnam/ns/`.

Proxim ORiNOCO 11b client PC card specifications, 2004.

P. A. Chou, Y. Wu, and K. Jain. Practical network coding. In *41st Allerton Conf. Communication, Control and Computing*, Oct. 2003.

D. da Hora, D. F. Macedo, L. B. Oliveira, I. G. Siqueira, A. A. F. Loureiro, J. M. Nogueira, and G. Pujolle. Enhancing peer-to-peer content discovery techniques over mobile ad-hoc networks. *COMCOM*, 2009. doi: 10.1016/j.comcom.2009.04.005.

M. Frodigh, P. Johansson, and P. Larsson. Wireless ad-hoc networking: The art of networking without a network. *Ericsson Review*, 4:248–263, 2000.

V. K. Goyal. Multiple description coding: Compression meets the network. *IEEE Signal Proc Mag*, 18(5):74 – 93, September 2001.

V. K. Goyal, J. Kovacevic, and M. Vetterli. Quantized frame expansions as source-channel codes for erasure channels. In *Proc. IEEE Data Compression Conf*, pages 326–335, 1999.

L. Kleinrock and J. Silvester. Optimum transmission radii for packet radio networks or why six is a magic number. In *NTC '78: National Telecommunications Conference*. IEEE, 1978.

T. Kunz. Multicasting in mobile ad-hoc networks: achieving high packet delivery ratios. In *CASCON '03*, pages 156–170. IBM Press, 2003.

S. Mao, X. Cheng, Y. T. Hou, H. D. Sherali, and J. H. Reed. On joint routing and server selection for MD video streaming in ad-hoc networks. *IEEE T Wirel Commun*, 6(1), 2007.

M. K. Marina, G. D. Kondylis, and U. C. Kozat. RBRP: A robust broadcast reservation protocol for mobile ad-hoc networks. In *Intern. Conf. on Comm.* IEEE, 2001.

S. Palchaudhuri, J.-Y. Le Boudec, and M. Vojnovi. Perfect simulations for random trip mobility models. In *ANSS'05: Annual Simulation Symposium*. IEEE, 2005.

E. Setton. *Congestion-Aware Video Streaming over Peer-to-Peer Networks*. PhD thesis, Stanford University, December 2007.

E. Setton, T. Yoo, X. Zhu, A. Goldsmith, and B. Girod. Cross-layer design of ad-hoc networks for real-time video streaming. *IEEE Wirel Commun*, 12(4):59–65, August 2005.

C. Tillier, T. Petrişor, and B. Pesquet-Popescu. A motion-compensated overcomplete temporal decomposition for multiple description scalable video coding. *EURASIP J Im Vid Proc*, 2007.

J. Tourrilhes. Robust broadcast: Improving the reliability of broadcast transmissions on CSMA/CA. In *PIMRC '98: International Symposium on Personal, Indoor and Mobile Radio Communications*. IEEE, 1998.

V.A. Vaishampayan. Design of multiple description scalar quantizers. *IEEE Transactions on Information Theory*, 39(3):821–834, May 1993.

M. Waldvogel and R. Rinaldi. Efficient topology-aware overlay network. *ACM CCR*, 33(1): 101–106, 2003.

Y. Wang, M.T. Orchard, and A.R. Reibman. Multiple description image coding for noisy channels by pairing transform coefficients. In *Proc. IEEE Workshop on Multimedia Signal Processing*, pages 419–424, 1997.

B. Williams and T. Camp. Comparison of broadcasting techniques for mobile ad-hoc networks. In *ISMAHNC '02: International Symposium on Mobile Ad Hoc Networking & Computing*. ACM, 2002.