Information Dissemination in VANETs based upon a Tree Topology

Patricia Ruiz[a], Bernabé Dorronsoro[b], Pascal Bouvry[a], Lorenzo Tardón[c]

[a] *University of Luxembourg, Luxembourg*
[b] *Interdisciplinary Centre for Security, Reliability and Trust, Luxembourg*
[c] *University of Málaga, Spain*

Abstract

In this work, we focus on vehicular ad hoc networks, also called VANETs, which are communication networks that devices (in this case vehicles) use to exchange messages in a distributed fashion, i.e., using no preexisting infrastructure. We first assess the feasibility of relying on a tree-based topology management structure for mobile ad hoc networks and in particular for VANETs. Next, we enhance DAGRS, an existing decentralized model for enabling distributed tree management and build BODYF on it, an efficient broadcast algorithm. Several broadcasting algorithms of the state of the art are implemented in order to compare the performance of BODYF. The approach is validated by simulation through three realistic scenarios located in Luxembourg city: the city center for both pedestrian and vehicles, and a highway environment. The comparison is made in terms of the coverage achieved by the broadcasting process as well as the complexity of the messages. DAGRS/BODYF approach outperforms other existing protocols in terms of both the number of devices reached and the network use.

Keywords: Vehicular ad hoc networks, distributed algorithms, tree topology, broadcasting algorithms

Email addresses: patricia.ruiz@uni.lu (Patricia Ruiz), bernabe.dorronsoro@uni.lu (Bernabé Dorronsoro), pascal.bouvry@uni.lu (Pascal Bouvry), lorenzo@ic.uma.es (Lorenzo Tardón)

1. Introduction

VANETs (or Vehicular Ad Hoc Networks) are vehicular communication networks which are spontaneously and automatically created among neighboring vehicles, without using any infrastructure —like routers, servers, etc. Hence, vehicles can communicate with other neighboring ones, that is car-to-car communication (C2C), or even with some existing infrastructure (C2I) like road side units. The main interest in VANETs is the possibility of developing applications that promise to make our driving safer, more efficient and more comfortable. This includes warning applications alerting other vehicles of a danger ahead, weather or traffic conditions, multimedia files exchanging between vehicles (for instance, music, movies) and/or some road side units (TV, radio, or news), etc.

However, there are also many challenging aspects that need to be solved before VANETs become a reality, since providing stable communications and a high quality of service (QoS) in these networks is still an unsolved problem. This undesirable behavior is due to the highly fluctuating topology of this kind of networks , caused by the fast speed in which vehicles are moving, the continuous appearance/disappearance of devices, communication disruptions, and also the unpredictable behavior of the network topology. Additionally, VANETs also inherit all the problems derived from the wireless networks, like the limited communication ranges, interferences, information loss, or signal attenuation due to obstacles.

Another important difficulty we must deal with in VANETs is the network partitioning. Due to the limited communication range of vehicles, it is common to have different separate clusters in the network which are not connected. Therefore, there might exist isolated nodes or groups of nodes that will not receive any information or message from any device outside this partition. For vehicular applications, when a device alerts about a danger ahead, e. g., a car accident, the warning message should reach as many devices as possible, thus all drivers are aware of it. As a consequence, there is no guarantee that a path always exists between any couple of vehicles. Hence, overcoming network partitioning is also a hot research topic.

The real development of efficient and reliable communication algorithms is very difficult in such high speed vehicular environment with a frequently changing topology. VANETs are, therefore, a major research topic covered by many national and international projects [1, 2] that involves not only academia but also car manufacturers and governmental institutions.

For disseminating information, broadcasting protocols play an important role in communications since many high level applications and even other protocols assume the existence of a broadcast service. However, the design of new efficient broadcasting algorithms for VANETs is still an open issue. Indeed, the importance of the good behavior of broadcasting algorithms leads many researchers to focus on optimizing the behavior of these dissemination protocols, e.g., maximizing the number of nodes reached and minimizing both the time required and the network overload [3]. This kind of protocols should reach as many devices as possible, even including devices in other partitions if the network is (as it is usually the case) partitioned, but at the same time reducing to the minimum the use of both the network and the device resources.

We are considering in this study a tree based topology in the network for the broadcasting process. Previous researchers demonstrated the validity of using spanning trees in networking area. Indeed, establishing a spanning tree in the network is a well known strategy for providing efficient communication and routing algorithms in wired networks [4], [5]. Furthermore, it is also a recent tendency to use them in mobile ad hoc delay tolerant networks (DTN) [6, 7, 8, 9, 10]. Due to the high speed of devices and hence, the changeable topology, the suitability of using a tree in VANETs may be called into question. Therefore, in this work we also present a comparison between the resources needed for maintaining the tree topology in a mobile ad hoc network (MANET) or in a VANET, since as we said before, it is extensively used and accepted in MANETs. The work here is not related to DTNs but VANETs and MANETs.

Building a tree topology over a network is a well known technique where the information the tree adds over the system can be really useful for many different applications like routing, search, etc. Moreover, in mobile networks where devices are moving and the topology is fluctuant, it is possible to create a tree where the nodes chosen to conform the tree are the ones with more stable links [7]. Therefore, for any critical application, using this kind of structures will provide more reliable results.

In this work, we extend the preliminary study in [9] in which an efficient and competitive broadcasting algorithm called BODYF was presented. BODYF was specifically designed for dealing with highly fluctuant networks as VANETs. In this kind of networks (usually very partitioned) BODYF outperformes the compared broadcasting algorithms. Besides the good performance of BODYF, one of its main important features is that it does not

require any parameter, in contrast to the main existing broadcasting algorithms in the literature, which need to be tuned for every different scenario in which they are applied [3].

BODYF is compared to some different broadcasting algorithms for VANETs present in the literature: (1) DFCN [11] that decides to resend a message in terms of the neighbor knowledge. (2) SAPF [12] that adaptively regulates the rebroadcast probability in terms of the speed of the device, (3) WPB [13] that calculates the forwarding probability in terms of the distance between the source and the receiver nodes, and (4) SF [14], since typically this protocol is not able to reach different partitions to the one wherein the broadcasting process was started, but it reaches instantly all devices in the original partition. Therefore, SF will be used as a reference of the starting partition size.

For our experiments, we use three very realistic scenarios taken from Luxembourg maps, i.e., the center of Luxembourg city for both pedestrian and vehicles, and a highway scenario we have defined for this paper. Each of these scenarios is studied with 3 different densities.

The main contributions of this paper are: (1) A comparative analysis of performance and stability features of tree based virtual topologies in VANETS and MANETS is proposed. To the very best of our knowledge this has not been addressed before. (2) An improved version of the implementation of DAGRS is presented, which directly impacts on the efficiency of BODYF increasing the coverage reached as result and reducing the communication overhead. (3) Several broadcasting algorithms from the literature were implemented in order to compare the performance of BODYF with these four different well known protocols. (4) Finally, the simulation environments for analyzing the behavior of the compared broadcasting algorithms have been redesigned. The networks used in this study are denser and new scenarios have been proposed. As a result, we are considering 9 different scenarios for comparing these 5 broadcasting protocols.

The rest of this document is organized as follows: We first summarize in Sect. 2 some of the most important broadcasting algorithms in VANETs. Section 3 describes BODYF, and the new improvements. Section 4 introduces DFCN, SAPF, WPB and SF, the broadcasting protocols compared to BODYF. Later, both the mobility models and the simulator used are presented in Sect. 5. After that, a study of the behavior of trees in both MANET and VANET is given in Sect. 6. The results of the comparison of the five studied protocols are shown in Sect. 7 and finally, Sect. 8 concludes

the paper.

2. Related Work

As we said before, VANETs are a hot research topic and have attracted a lot of attention in the last years. Moreover, broadcasting is a key feature for networking, but it has a special role in VANETs where researchers try to investigate the possibility of making driving safer. For that, disseminating a message alerting vehicles of a danger ahead on the road is a must. In this section we present some of the broadcasting algorithms for VANETs that already exist in the literature.

There are mainly two different variants of broadcasting algorithms in VANETs: (1) those aimed at disseminating an alert of a danger ahead backwards (to vehicles approaching in the same lane); (2) spreading the message in any direction as there might be different vehicles interested in this message. Some existing protocols in the literature of both tendencies are explained below.

2.1. Disseminating the message backwards

The main goal of this kind of broadcasting algorithms is to alert drivers from an existing accident preventing new possible collisions. Vehicles approaching the area (in the same lane) where the accident occurred are aware of it and thus, reduce the speed avoiding a new possible accident. In [15] Da Li et al. proposed a distance-based directional broadcast (EDB) for VANET using directional antennas. Only the furthest receiver is responsible of forwarding the packet in the opposite direction where the packet arrives. Moreover, there are repeaters located at intersections to help disseminating the data.

Tonguz et al. present in [16, 17] a broadcasting algorithm that depends only on the local topology information and that is robust against different types of vehicular traffic conditions. It handles both, the disconnected network and the broadcast storm problem [18] in a completely distributed fashion. The main criterion to determine how to handle the rebroadcast is based on the list of one hop neighbors.

A vehicular multi-hop broadcasting protocol (VMP) for fast dissemination was introduced in [19]. VMP designates multiple forwarding nodes with different delays to disseminate an alert in a concrete area. To ensure high

reachability a cooperative forwarding mechanism is used and also, a duplicated packet detection procedure is provided to discard unnecessary rebroadcasts.

2.2. Disseminating the message in any direction

It is known that usually when an accident occurs, there is a high probability of collision in the same area of cars approaching in opposite direction as drivers tend to look at happened. Therefore, disseminating the message in any direction.

Broadcasting algorithms that do not focus on a concrete direction can be used not only for alerting vehicles of a danger ahead, but also for alerting a closer base station about the accident or for changing the route in order to avoid a traffic jam, or just to help routing algorithms to find the destination nodes.

In [20] a broadcasting protocol that decides the probability of rebroadcasting the message in terms of the neighbor knowledge was introduced, DFCN. The source node includes in the message a list with its one hop neighbors, thus the receiver will calculate the number of its neighbors that are not included in the list and decides to resend to this value. This algorithm not only works for VANETs, but also for MANETs after modifying some parameters.

In vehicular ad hoc networks different protocols based on probabilistic schemes have been proposed. In [13] three distributed probabilistic and timer-based broadcast suppression techniques were presented: weighted p-persistence, slotted 1-persistence, and slotted p-persistence schemes. The proposed schemes rely on GPS information (or received signal strength when a vehicle cannot receive a GPS signal). Denoting the relative distance between the source and the receiver nodes and the average transmission range, the forwarding probability, can be calculated assigning higher probability to nodes that are located further.

Also based in a probabilistic scheme, [21] presents a *nth* power p-persistent broadcasting protocol for dense vehicular ad hoc networks. The basis are the same as the weighted p-persistence in [13], but the probability is elevated to the *nth* power. This proposal is proved to be efficient in very dense networks.

Slavik proposed in [22] another probabilistic scheme that is anonymous and scalable. The main goal of this protocol is the privacy. The driver is not willing to adopt a technology that allows third parties to monitor their movements, so that no information between drivers can be exchanged. Two approaches are performed: (1) nodes are given a uniform and constant

retransmission probability, and (2) the retransmission probability is dynamically determined based on the distance between the receiver node and the last hop.

A speed adaptive probabilistic algorithm was proposed in [12], SAPF. The rebroadcast probability is adaptively regulated, based on the vehicle speed, to optimally reduce message delivery delays caused by increased contention, in areas with high density of vehicles. The goal of this work was to find a relation between the speed of the device and the forwarding probability that finds the optimal value for the latter in terms of the former.

In [23], an opportunistic relay scheme for cooperative collision warning in VANET, which helps to mitigate the impact of transmission failure caused by shadow fading was presented. Devices periodically exchange motion information with their neighbors. Each node not only calculates its own collision risk but also its neighbors', so that if a local vehicle has detected a danger and the neighbors keep quiet or reply wrong responses, the vehicle may generate a relay packet that contains the motion information of the two vehicles involved in the danger and broadcast it.

3. Broadcasting Over Dynamic Forest, BODYF

BODYF is an efficient broadcasting protocol specifically designed for highly dynamic ad hoc communication networks (as VANETs). It was originally presented in [9], and in this paper we propose an improved version, more efficient, in which communications between vehicles have been improved as it will be explained in subsection 3.2. One of the main features of BODYF is the absence of parameters in its configuration, such that no tune is needed for the different scenarios in which it could be used. This outstanding characteristic distinguishes it from other well-known broadcasting algorithms that typically have several parameters to tune impacting in its performance [3]. Although it has no parameters, BODYF relies on a tree topology that might have some. For example, in this work we use DAGRS that depends on the *hello* interval and the speed of the token.

One classical way to enable service on mobile ad hoc networks consists in establishing a topology management structure such as a dominant sequence [24] or a tree [25]. This structure is maintained during the whole lifecycle of the ad hoc network using the exchange of beacon messages and minimal information stored at the level of the nodes. Higher level services such as routing then rely on the underlying structure. BODYF relies on a spanning

forest. Our main goal is not the tree itself, but the design of a new broadcasting protocol that achieves the best possible coverage and network use at a minimum cost, using the information of the network provided by the tree-based topology. We assume that it can be already present in the network, since it is commonly used for routing purposes or any other necessity.

We present in Sect. 3.1 DAGRS, the decentralized model we use for creating the tree-based topology in the network, as well as the implementation we designed for VANETs. After that, BODYF is presented in Sect. 3.2.

3.1. Dynamicity Aware Graph Relabelling Systems, DAGRS

DAGRS (or Dynamicity Aware Graph Relabelling Systems) [25] is a completely decentralized model for building tree-based topologies in dynamic graphs. The model only uses information about direct neighbors for building the topology. Hence, if we represent the VANET as a graph, where vehicles are the set of vertices *(V)*, and the links between them are the edges of the graph, *(E)*, then we can use DAGRS to build the desired tree-topology in our network. The dynamicity of the network is represented by the fact that both V and E can change at any time.

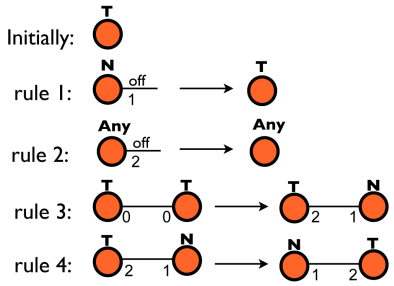


Figure 1: DAGRS rules for creating spanning forest topologies.

A tree in a graph is, by definition, a connected cycled-free subgraph, and a forest is defined as a set of trees. The objective of DAGRS is to build a spanning tree on the graph, but this is typically not feasible due to network partitioning and devices mobility, so DAGRS will be used to build a forest topology. This is operated by locally applying some simple rules in every node, as shown in Figure 1. T represents a node with token, N is a device without token, and *Any* means it can be any of them. The numbers on the edges are labels representing the route to the token. A token does not have any information related to the tree, it just provides a status to a node that will allow the tree merging process. By using a single token per tree mechanism,

the formation of cycles is not possible, as only the node possessing the token is able to merge its tree with another one. The 4 possible rules defined in DAGRS are:

- rule 1: A tree link breaks, and the node belongs to the sub-tree which does not possess the token (indicated by the label on the edge). In this case the node must regenerate the token, otherwise there will exist a tree without a token (which is an undesirable situation).
- rule 2: A tree link breaks, and the broken link occurs at a node which currently belongs to the sub-tree which possesses the token. In this case, the node does nothing regarding the maintenance of the token.
- rule 3: When a node with token meets another device possessing a token; both nodes will try to merge their trees in order to obtain a bigger tree from the two existing ones. The trees merging process starts. As result of this rule, a bigger tree and only one token remain (the merging process discards one token automatically in order to have one and only one token within each tree).
- rule 4: Token traversal in general case: the token visits the nodes of the tree following a given strategy.

Initially, all devices are labelled T, which means they all constitute one-node trees. The algorithm then, performs on the basis of the four rules described above to generate the spanning forest topology in the dynamic network. It is important to emphasize that DAGRS model itself does not model services or applications, it simply models the mechanisms to handle with topology changes and interaction between devices. In this model we only use one-hop neighbors information, so it is a localized algorithm.

In the case of VANETs, given that we are dealing with distributed systems, it is necessary to exchange some messages between nodes for merging trees and also for circulating the token. When a device receives the token, it should check if there is any neighbor holding another token around, but this behavior would lead to a massive interchange of data in the network. In our implementation, as the protocol aims to be specifically designed for highly fluctuating topology, we benefit from the high mobility: only when a change in the neighborhood is detected (since the last time the device had the token), the node sends a message to check if there is any neighbor with

token around. We consider that there is a change in the neighborhood when a new neighbor arrives or an old one leaves. These changes are detected using the beacons (*hello* messages).

Now, let us explain the message exchange that is produced when two neighbors holding tokens meet. We can imagine node *'A'* receives the token and also detects a change in its neighborhood. In this situation, it will send a broadcast message to its one hop neighbors to notify that it has the token, and it waits for an answer of any neighboring device saying it also has the token. Due to the mobility and the high speed of the devices, in our implementation the first node answering will be the selected device for merging trees. When the node sends this broadcast message it also sets a timer, if it does not receive any answer during 0.3 seconds, the node will circulate the token. This time was experimentally chosen [9], and it ensures a neighboring device is able to receive the broadcast message and send an answer (in case it has the token) before the timer expires.

When *'A'* receives the answer of a neighboring device holding a token, the merging process starts. During this process one of the tokens must be deleted. In DAGRS, any of the two tokens are candidates to be deleted, but in a distributed system a procedure ensuring that only one of token is deleted must be implemented. In our implementation, the node with higher address will be the one deleting the token.

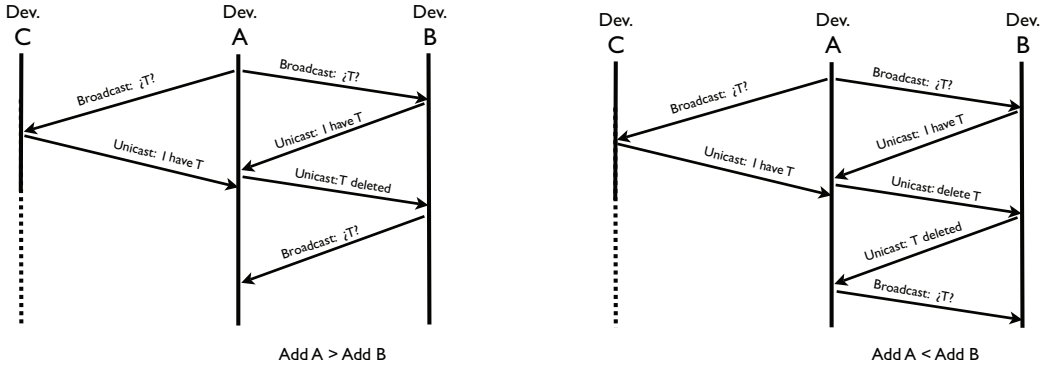


Figure 2: Messages exchanged for creating a tree topology in the network.

In Fig. 2, we show the messages exchanged between two nodes for merging their corresponding trees. In the left diagram, node *'A'* is deleting the token, meanwhile in the one on the right it will be *'B'*. As it was explained before, the first node answering will be selected for merging their trees (in both

diagrams it is node *'B'*). When node *'A'* receives the message from *'B'*, it checks which one has the highest address. In case it is *'A'* (left figure), it will first create the logical link between them, delete the token and send a message to *'B'* informing it about the deletion of its token. In case *'B'* has the highest address, node *'A'* sends a message requesting the token deletion and node *'B'* will send a confirmation message after deleting it. When node *'A'* receives the token deletion confirmation, it creates the logical link between them. In both cases, the node that finally possesses the token sends a broadcast message again asking if there is any neighbor with token around.

Figure 3: Steps in the creation of a tree topology in the network.

In Figure 3, we show the construction of a tree in different stages. The first step is reflecting the initial situation where all devices possess the token. Then, to merge their trees, a node (the red one in step 2) broadcasts a message (represented by a circle) asking if there is any node possessing a token in its range. All nodes with token answer to it by unicast (represented by an arrow). The first node answering is the one chosen for merging. In step 5 and 6, the messages for merging trees and token deletion are shown, and finally in step 7 the tree is formed (notice that there is only one token in the new tree, the other one was deleted). The same procedure starts again in step 8 in order to merge other nodes into one single tree.

The token is traversing the tree following a Depth First Search (DFS). Every node has a list with the one hop neighborhood, so we use this list for circulating the token. As we are dealing with distributed systems, it is

necessary to keep track of the node that sent the token to the current device (called *upper neighbor*) and also about the nodes that were already visited. In that way, the device is sending the token to all its neighbors. When the token is received from the neighbor the current device sent it to, it will forward it to the next neighbor in the list. Once the list is finished, it means that the token was already sent to all neighbors, it is then sent back to the *upper neighbor*. In previous work [26], different strategies for traversing the token were compared in terms of both the number of trees conforming the spanning forest and the speed to build them. The different strategies studied for traversing the token in a tree topology were DFS, Tabu (the token keeps a list of forbidden neighbors), and random (the token is sent to a randomly chosen neighbor). They were compared in terms of the speed of convergence, that is, how fast a connected component converges into one tree, and also in terms of the performance ratio, that is the number of trees in each connected component. It was concluded that DFS behaves better for both metrics, we therefore adopted it for our experiments. Notice that in this distributed implementation, there is no notion of *root of the tree* as there is no knowledge of the global tree locally in each device. The only similar concept is the one used for DFS, the *upper neighbor*, where a difference is made between this node and the rest of logical neighbors (that can be considered as children). But this differentiation is only taken into account for traversing the token.

In Figure 4, the mechanism of DFS is shown. Considering node A sends the token to node B for the first time, node B considers node A as *upper neighbor*. Because node B does not have any other neighbor but node A, it sends the token back to its *upper neighbor* (A in this situation). Next, after node A receives the token from node B, it sends the token to the next neighbor in the one hop neighbor list, that is, node C. Similarly to node B, node C considers node A as *upper neighbor*, and traverses the token among its other neighbors: nodes E and F (for these two nodes C is the *upper neighbor*). When the last neighbor, node F in this case, returns the token, node C sends the token to node A, as it is considered *Cś upper neighbor*. If node A has no more neighbors and no *upper neighbor* (as shown in the example), node A will start the same procedure. This notion of *upper neighbor* is only considered for traversing the token.

The improvements that the new version of BODYF provides are not included in the broadcasting protocol itself, but in the communication message exchanged for conforming the tree topology. In the previous version of BODYF, once the tree merging process of DAGRS was finished, the node

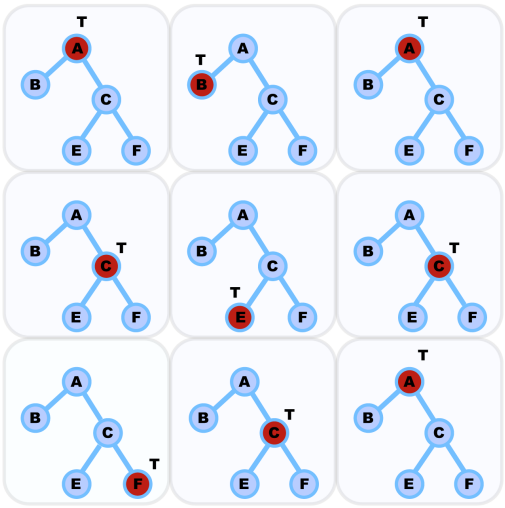


Figure 4: The DFS token traversal strategy.

keeping the token just traversed it. In that way, if a node detects two new neighbors holding token and starts the merging process with one of them, the merger with the other is not performing until another change in the neighborhood is detected (the speed of convergence is slower). In this situation, the system only performs well in very dynamic networks, but in static networks it does not. In our new implementation, after the merging process, the node with the token will ask if there exists any other neighbor holding a token. If there is no answer, the token is traversed, but if a node answers, the merging process will start again. Therefore, the performance of creating the tree is improved. Now, with this new improvement, BODYF performs well in both cases static and dynamic networks.

A second difference between the previous and the new version is the number of messages sent while merging two trees. In the original implementation if we consider Figure 2, node *'B'* always sent a message to node *'A'* confirming the deletion of the token or if node *'A'* deleted it, just confirming the merger. The new version, see Figure 2, avoids this message in case that node *'A'* deletes the message since, if node *'B'* goes and the merging process does not finish, *'A'* will detect it using rule 1 in Figure 1. So in this version the number of messages needed to create the tree is reduced.

Algorithm 1 Pseudocode of BODYF.

Data: m: the incoming broadcast message.
Data: d: the node receiving broadcast message.
Data: s: the node which sent m.

```
1: if m is received for the first time then
2:     if s and d are logical neighbors then
3:         d → forward m;
4:     else
5:         wait until the token is received → d possesses the token;
6:         if d received m also from its tree then
7:             d → discard m;
8:         else
9:             d → forward m;
10:        end if
11:    end if
12: else
13:    d → drop m;
14: end if
```

3.2. The BODYF Protocol

Assuming we have a forest topology in our network, we will use it to disseminate information using BODYF. BODYF distinguishes between two kind of neighbors in the network: (1) *logical neighbors* are those ones belonging to the same tree, while (2) *potential neighbors* are those that do not belong to the same tree yet but are in communication range, as it is shown in Fig. 5. In this picture, devices in same color belong to the same tree. The links between devices are represented as a continuous line if the neighbors belong to the same tree (logical neighbors), or as a dashed line if they are either in communication range but do not belong to the same tree or belonging to the same tree but not connected in order to avoid cycles (potential neighbors).

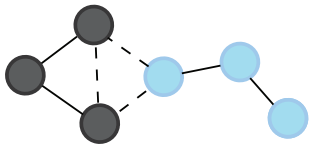


Figure 5: Possible types of neighbors in a spanning forest algorithm.

The functioning of BODYF is described in Algorithm 1. We suppose that every message has a unique identifier, and when a device receives the same message more than once, it will directly discard it (no processing is done). When a device wants to spread a message, it will broadcast the information

to its neighborhood. We could think on doing a multicast only to its logical neighbors (the neighbors in the same tree), but this operation supposes the same network load as the broadcast in wireless communications (or even less if we consider the list of addresses included in the multicast), and the latter is allowing the potential neighbors (not belonging to the tree) to also receive the message. When a device receives the message from a logical neighbor, it will forward it only if it was received for the first time (lines 1-3), otherwise it is dropped (lines 12-14). In the case it was received from a potential neighbor, the device will keep it until it receives the token (line 5). Once the device possesses the token, it will forward the message just in the case that it did not receive the same message from a neighbor belonging to its tree during the time it was waiting for the token (lines 6-10). That is the way the message can spread through different trees, trying to avoid the dissemination of the a message more than once in the same tree. Additionally, the delay introduced when a device receives the message from a potential neighbor helps to spread it to other partitions of the network, thanks to the devices mobility.

4. The Compared Protocols

In this section we give a brief introduction to the other protocols we are using for the sake of comparison: simple flooding (SF) [14], delayed flooding with cumulative neighborhood (DFCN) [11], speed adaptive probabilistic flooding (SAPF) [12], and weighted p-persistent broadcasting (WPB) [13].

4.1. Simple flooding, SF

We decide to use SF for the comparison because it reaches instantaneously all devices inside a partition, but it is not efficient at all regarding the bandwidth, and it is not able to spread the message outside the partition where the source node is. Hence, it provides a good approximation to the initial partition size, and it will show the validity of the other compared protocols. In connected networks the coverage it achieves can be seen as an upper bound. SF is the most intuitive idea for disseminating a message in a network. It does not try to reduce the number of re-emissions or collisions, so it does not need any knowledge about the neighborhood. The strategy of this algorithm is quite simple, when a device receives a message, it will forward it only once.

4.2. Delayed flooding with cumulative neighborhood, DFCN

DFCN [3, 11, 27] was chosen because it was demonstrated to outperform some other well known broadcasting protocols in the literature as Simple Flooding, Flooding with Self-Pruning, the Scalable Broadcast Algorithm, Multipoint Relaying and AHBP-EX [27, 20], and because it was also dealing with high mobility (DFCN was used in three different scenarios shopping mall, metropolitan area and highway). DFCN was designed with the aim of minimizing the network overload (by suppressing some rebroadcast). For doing that, the node sending the message embeds a list with its one hop neighborhood. The receiver node will decide whether forwarding the message or not according to the *benefit* of an operation, measured in terms of the number of its one hop neighbors that are not included in the list (i.e., the number of nodes that most probably did not receive the message). For avoiding collisions, every device receiving a message sets a random delay (RAD). This way, two nodes in the same neighborhood forwarding the broadcasted message will probably not resend it at the same time. However, in order to promote the dissemination of the message in sparse networks, every time a new neighbor is met, RAD is fixed to zero. The ability to know when a network is sparse or not is directly related to the number of neighbors of a given node, and defined by *densityThreshold*. As we mentioned before, DFCN has 3 parameters to set:

- The *benefit*: is a fixed threshold in order to decide if a message should be forwarded or not.
- The RAD: the period of time a node waits before forwarding the message. It is a random value chosen from a fixed interval.
- The *densityThreshold*: is a threshold for considering if the network is dense or sparse. It is directly related to the one hop neighborhood.

4.3. Speed Adaptive probabilistic flooding, SAPF

SAPF [12] is a probabilistic broadcasting algorithm that adapts the rebroadcasting probability to the speed of the current device. One of the main challenges of this dissemination algorithm is the selection of a rebroadcast probability that achieves low delay and high reachability. A high forwarding probability for emergency messages will reach more devices but it will also increase both the network resources and the delay due to the higher amount

of rebroadcasting vehicles. In this work, the critical forwarding probability is related to the network density which in turn depends on the vehicle speed (low speed implies high density). For obtaining optimal performance of the broadcasting protocol at all network densities, the forwarding probability dynamically adapts to the speed of the device. Whether the speed is either too high or too slow, the probability chosen is fixed. In case low speed is detected, the fixed rebroadcast probability value is low, as there might be a traffic jam. On the contrary, if network density is low (high speed), the vehicles always rebroadcast in other to reach as many devices as possible. The pseudocode for the algorithm is shown in 2.

Algorithm 2 Pseudocode of SAPF.

Data: v = vehicle speed
Data: m = broadcast message

1: **if** (v >= 15km)&&(v<= 100km) **then**
2: broadcast m with probability **p = 0.00557*v*-0.0033**
3: **else**
4: **if** v< 15km **then**
5: broadcast m with probability **p = 0.05**
6: **end if**
7: **else**
8: **if** v > 100km **then**
9: broadcast m with probability **p = 1**
10: **end if**
11: **end if**

On one hand if the speed of the vehicle is very low, probably means there is a traffic jam, and therefore, a low broadcasting probability is used as the network is almost static and a high probability would lead to the broadcast storm problem [18]. On the other hand, when the speed is very high, the network is very fluctuant and the connectivity between devices is difficult, therefore the forwarding probability is set to 1 in order to promote a high reachability.

4.4. Weighed p-persistent broadcasting, WPB

WPB [13] is also a probabilistic scheme that calculates the forwarding probability in terms of the distance of the current (*node i*) to the source node (s), see Equation. 1.

The first time a node receives a packet sets a *WAIT_TIME*. During this *WAIT_TIME* the distance to the new sources of all the repeated messages received is calculated, and thus, the probability of resending. The smallest probability is the one selected as its forwarding probability.

$$probability_i = \frac{distance_{is}}{transmission\ range_i} \quad (1)$$

If the message is not rebroadcast, the node stores the message for an additional $WAIT\_TIME + \delta$ ms, where δ is the one-hop transmission and propagation delay, which is typically less than $WAIT\_TIME$. If the node does not hear the retransmission of the message from any of its neighbors the node should rebroadcast the message with probability 1 after $WAIT\_TIME + \delta$ ms in order to prevent message die out and guarantee 100 percent reachability. In weighted p-persistence the further the location the higher the probability of resending.

5. JANE Simulator and the Proposed Mobility Models

In this work, we evaluate BODYF and compare it to DFCN, SAPF, WPB and SF in three different scenarios: The first proposed scenario is a pedestrian area of a city center. In this case, we are considering the city center of Luxembourg (see Figure 6), that was presented in [28], but it is applicable to any city center and therefore, to any MANET. The second scenario considered is a VANET created in the city, (see Figure 7) and finally the third one is a VANET in a highway (see Figure 8).

The main difference between the two scenarios dealing with vehicles is the speed of the devices and, more important, the number of crossings in the path. In the highway there are just a few exits where cars have to reduce the speed, and entrances where devices go slower at the beginning while incorporating into the highway. However, in the city, there are plenty of crossroads where devices reduce their velocity or even stop due to traffic lights or pedestrian crossing and the speed limit is considerably slower than in the highway.

As we are considering three different scenarios: (1) the pedestrian city center, (2) the vehicular city center and (3) the highway, the density of the network can not be measured as a fixed number of devices for all of them since the environments are completely different. In order to have a balanced density in all environments, we consider the total length of all available paths in the different scenarios studied, and set the density of the network according to the number of nodes per meter. We measure the total length of all possible paths in each scenario and we obtained: (1) 4473 m in the pedestrian city center, (2) 13250 m in the vehicular city center and (3) 6150 m in the highway.

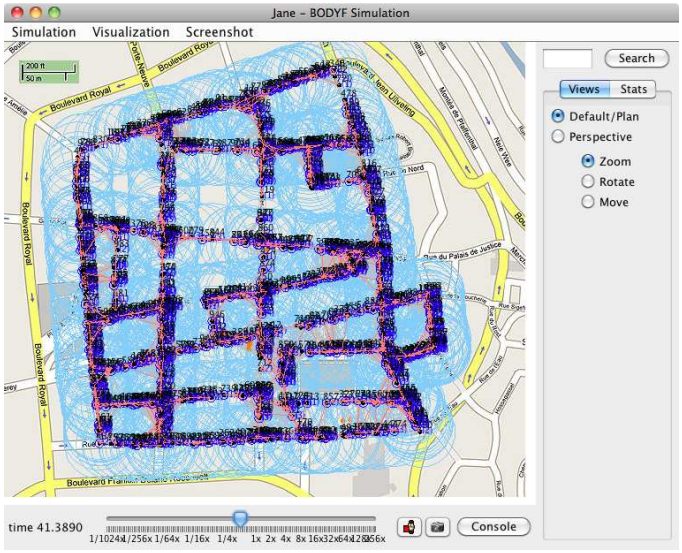


Figure 6: Pedestrian area MANET.

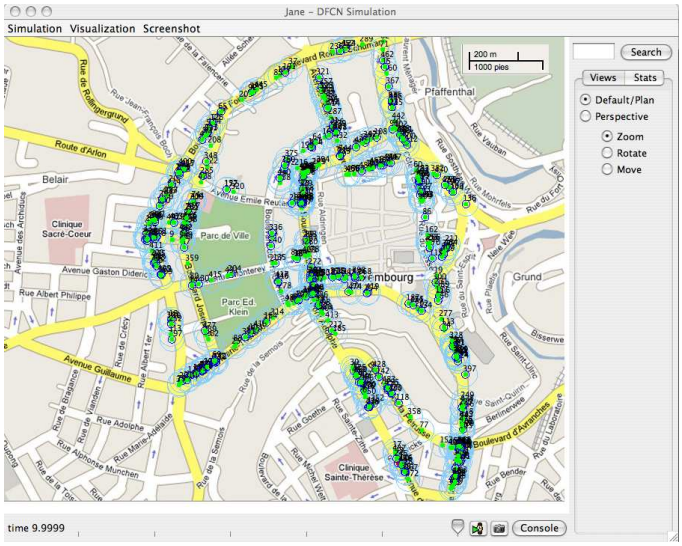


Figure 7: Vehicular city center.

For our comparison experiments we are considering three network densities: (1) a device every 25 m, (2) every 15 m and (3) every 10 m. The different configurations obtained for our experiments are presented in Table 1.

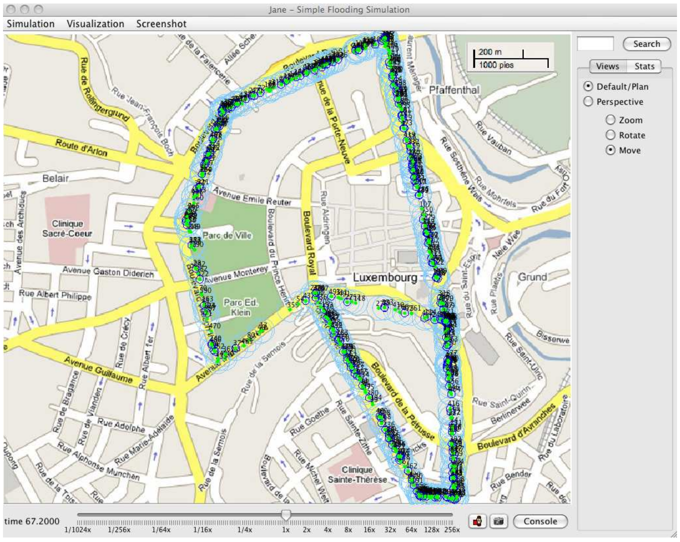


Figure 8: Highway.

Table 1: Configuration parameters.

		MANET	Vehicular Center	Highway
	Length	$4473m$	$13250m$	$6150m$
Densities (devices)	**25 m/dev**	179	530	246
	15 m/dev	298	884	410
	10 m/dev	447	1325	615
	Coverage	$50m$	$250m$	$250m$
	Speed	$0.5 - 2m/s$	$9 - 21m/s$	$21 - 36m/s$

As shown in Table 1 for simulating the pedestrian city center (MANET) we are considering that the speed can vary between 0.5-2 m/s, that is 1.8-7.2 Km/h and the transmission range of the devices considered is 50 m. For both, the highway and the vehicular city center scenarios the transmission range is 250 m, meanwhile the speed is different: 9-21 m/s in the city and 21-36 m/s in the highway (30-75 km/h and 75-130km/h, respectively).

We developed a mobility model where vehicles move along streets, stop at crossroads and also overtake each other. In the city center scenario the real senses of the lanes are kept, showing the most congested areas in reality. In the highway environment all roads have two senses. For the pedestrian model, the devices are also restricted to move only along the streets, but devices move in both senses.

The mobility model is random waypoint with restrictions. Devices can

only move from one crossroad to another. The next movement is randomly chosen from the small set of possibles destination considering the current position (the mobility is restricted to the possibles movements along the roads in Luxembourg).

The devices move on a straight line with a constant speed from one position to another. The mobility model uses a directed graph given as XML file for device movement. Vertices are crossroads and contain routing probabilities, all possible destinies have the same probability of being taken; the next route is chosen at random. Edges can have an arbitrary width so that devices move on a lane and not only on a strict line (allowing overtaking).

In these environments the diffusion process is more difficult than in the case of using other mobility models as random waypoint, since movements are more restricted but, for sure, the simulation is much more realistic. Our main goal is to get results as close as possible to a test on real devices moving along roads in a city center or a highway.

For simulating these scenarios we are using JANE simulator [29]. The main feature of JANE is its three steps development; it facilitates the evaluation and minimizes the effort needed for software development in MANETs, so that the evaluated code in the simulated environment can be directly executed in real scenarios without modifications. It is implemented in Java and it is an event driven simulator. This tool was jointly developed by the universities of Trier (Germany) and Luxembourg, and it is currently being used in several research projects.

6. Using Tree Topology in VANETs

As we said before, the use of spanning trees is widely studied in the literature for fixed, mobile ad hoc wireless networks, and even for delay tolerant networks (DTN), but establishing a tree topology in a VANET may lead us to a contradiction. On the one hand, the restricted mobility of devices, all moving one after the other with quite similar speeds (thus, a low relative speed) and bigger coverage range than MANETs make us thinking that is a quite good scenario for a tree topology. On the other hand, the high speed of devices, overtaking each other and meeting devices circulating in different directions provoke that some researchers do not think it is a proper environment for dealing with trees.

As having a tree topology working over MANETs is strongly accepted in the research community, we want to check by making realistic experiments if

it is suitable or not to use them in vehicular networks. For that, we are comparing the resources needed for maintaining a tree in MANETs and VANETs. In our experiments, we are considering the 3 different scenarios presented in Sect. 5 (see figures 6, 7 and 8), in order to see how the maintenance of the trees evolve in each situation.

In order to show the validity of using tree topologies over VANETs we want to study the stability of the tree in a network by studying the behavior of its links. For that, we are considering three different aspects of the network. The first aspect considered is a measure of the number of tree links created and broken during the simulation. The second aspect is the total number of messages exchanged for creating the tree. And finally, we are also considering the volatility of the tree links, what refers to the number of connections and disconnections a concrete link suffers during its life time. In the following subsections, we will explain and analyze in detail each of the aspects we are considering.

6.1. Created and broken links

For obtaining the number of links created and broken, we are running DAGRS in the three scenarios (with three different densities) for 630 seconds. The number of links created/broken obtained is strictly related to the number of devices in the network. Hence, it is necessary to normalize the data obtained by the total number of devices in each network. In Table 2 the total number of created or broken links per device during the simulation time (630 seconds) is shown.

Table 2: N. of created/broken tree links per device for 630 seconds.

	Densities	**MANET**	**Vehicular Center**	**Highway**
Created Links	**25 m/dev**	$10.6423_{\pm 0.2887}$	$12.7742_{\pm 0.2692}$	$16.5217_{\pm 0.4770}$
	15 m/dev	$10.8221_{\pm 0.2110}$	$11.6663_{\pm 0.1654}$	$14.8687_{\pm 0.3862}$
	10 m/dev	$10.5504_{\pm 0.1816}$	$9.6153_{\pm 0.133}$	$13.1442_{\pm 0.2702}$
Broken Links	**25 m/dev**	$9.7572_{\pm 0.2811}$	$11.8890_{\pm 0.2444}$	$15.7003_{\pm 0.4418}$
	15 m/dev	$9.9020_{\pm 0.20691}$	$10.7595_{\pm 0.1449}$	$14.0150_{\pm 0.3570}$
	10 m/dev	$9.6589_{\pm 0.1673}$	$8.7090_{\pm 0.1171}$	$12.2908_{\pm 0.2499}$
Ratio	**25 m/dev**	0.9168	0.9307	0.9502
	15 m/dev	0.9149	0.9222	0.9425
	10 m/dev	0.9155	0.9057	0.9351

We can observe in Table 2 that the behavior of the tree topology is quite similar in all the scenarios. Both, the number of created and broken links per device in 630 seconds in all scenarios do not differ much. The difference

in the number of tree links created in the three scenarios is small. The maximum difference is found with the lowest density between the MANET and the highway, and it implies that in the highway each device creates 6 more links in 630 seconds (1 more created link every 105 seconds). And the number of broken links differs in the same proportion. As the chance of breaking a tree link is higher when having more created links, we should consider a parameter that gives an idea of the stability of the links, therefore, we compute the ratio between the number of links broken and the number of links created. This ratio is also provided in Table 2. The smaller the ratio the more stable the topology (the closer to 0 the more stable). It is also possible to check that the value of this ratio is also quite similar in all the studied cases. The behavior between the MANET and the other two VANETs is similiar, the biggest difference in this ratio is found between the densest network in MANETs and sparsest network in the highway scenario (0.0347).

It is also necessary to remark that when the density of devices is the biggest, 1 device every 10 meters, the vehicular environment shows better results than the pedestrian city center and the highway scenarios. We can think after showing this phenomenon, that the behavior of the tree a MANET or a VANET is quite similar, and really depends on the characteristics of the network.

6.2. Number of messages exchanged

As we mentioned before we also took into account the total number of messages exchanged for the topology. That is, the number of broadcasted messages trying to merge trees, and the unicasts sent for the merging process. Therefore, all the messages exchanged for creating and maintaining the topology are considered. In Table 3, the bandwidth each device uses per second for that is shown. For estimating these data, we suppose we are dealing with IPv6, and the header of each message exchanged is 40 bytes. In our implementation of DAGRS the packets sent are empty, we just use the identifier. Therefore, we consider each message is 40 bytes. The total bandwidth used per device per second for the complete creation and maintenance of the tree during the 630 seconds of simulation is shown in Table 3.

We should also mention that the implementation we made for DAGRS and the configuration of the devices (transmission range) favor that the number of messages exchanged in VANETs is bigger than in MANETs. The reason is that we consider a node will only try to merge its tree with a

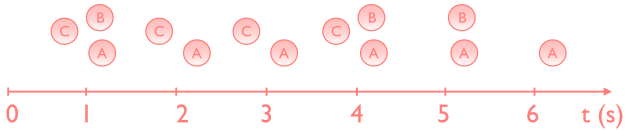
Table 3: Bandwidth used per device per second (bytes).

Densities	**MANET**	**Vehicular Center**	**Highway**
25 m/dev	$0.8189_{\pm 0.0119}$	$2.5504_{\pm 0.0914}$	$3.3104_{\pm 0.1380}$
15 m/dev	$0.6869_{\pm 0.0129}$	$4.7078_{\pm 0.1389}$	$5.4067_{\pm 0.1511}$
10 m/dev	$0.6748_{\pm 0.0131}$	$10.0682_{\pm 0.1981}$	$8.9581_{\pm 0.1981}$

neighbor (and hence sends a broadcast message), when a change in its neighborhood is detected or has just merged with another node. But since the transmission range for VANET is 250 m and 50 m for MANET, and also the mobility is higher in VANETs, devices in the vehicular network are much more sensitive to changes in the neighborhood than devices in MANETs (the neighborhood is 5 times bigger), so nodes send more broadcasting messages looking for a neighbor with token. However, even considering a much bigger neighborhood, the bandwidth each device uses in the worst case is 10.0682 bytes/s. We consider this is an acceptable rate, moreover if we consider all the advantages of having this topology over a network.

6.3. Volatility

The third aspect studied to compare the feasibility of using tree topologies in VANETs is the volatility of the tree links in all the scenarios studied. We consider the volatility of the link can be measured as the total number of appearances of a link between two devices according to the total time this link is alive. The volatility varies between 0 and 1. The closer to one, the more volatile, while the closer to 0 the more stable. This idea was taken from [30], where the volatility is presented as shown in Figure 9.



A-B	1-1	1-1	1-1	2-2	2-3	2-3
A-C	1-1	1-2	1-3	1-4	1-4	1-4
C-B	1-1	1-1	1-1	2-2	2-2	2-2

Figure 9: Process for measuring the volatility of a link.

As we can see, the information initially stored for all the links is the same: first time appearing in the network (1), and also the duration of the link (1).

In the second time stamp, node B disappears, so node A will increase the life time of link A-C, but not the link A-B since it disappeared. In the fourth time stamp, node B appears again, so that, both node A and C will increase the appearance of this link and also the life time. When the simulation time is finished, the volatility can be calculated as in Equation 2:

$$volatility\ of\ a\ link = \frac{number\ of\ appearances}{life\ time} \tag{2}$$

For measuring the volatility of the links it is necessary to discretize the time. In our experiments, for estimating the volatility we are running DAGRS for 600 seconds and every second each device stores local information about the current links it has with its neighbors: the number of appearances and the life time. We need to store the number of times each link appears and also for how long this specific link is on.

In Table 4, the average of the volatility of the whole network is presented. We calculated the volatility of each link during 600 seconds for the three scenarios studied (with each of the three densities) in this work, and computed the average. We can see the values of the volatility are very low, which means the links are quite stable. In the worst case, we have value 1, which means that as soon as the link appears, it disappears. But in fact, most of the links are connected more than 1 second, and as time passes, the volatility of the link decreases quickly (a link that appears only once and is ON for 10 seconds has a volatility of 0.1). The ideal network regarding the volatility is a static one, where all the links appear in the first second and do not disappear anymore. In this case, if we consider the same simulation time, the best value of the volatility we can obtain is 1 / 600, what means 0.0016667. As we can observe in Table 4, the volatility of a MANET is smaller than in the vehicular center, and this one is also smaller than in the highway. This is due to the speed the devices have in each scenario. Although the MANET has a smaller volatility, we assume that the volatility obtained for the scenarios dealing with cars is small enough, and therefore, the topology considered is stable. We can see that the volatility is hardly related to the density of the network (number of devices), since the values obtained for the volatility in each environment are quite similar despite the density.

Considering that the average of the volatility does not provide a complete view of the behavior of the topology regarding the stability along the time. We also computed the probability distribution of the volatility. We show in Figures 10 and 11 the probability of having a certain volatility. We calculate

Table 4: Average of the volatility of the links.

Densities	**MANET**	**Vehicular Center**	**Highway**
25 m/dev	$0.0467_{\pm 0.0704}$	$0.0679_{\pm 0.0868}$	$0.0989_{\pm 0.1121}$
15 m/dev	$0.0475_{\pm 0.0716}$	$0.0670_{\pm 0.0832}$	$0.0936_{\pm 0.1039}$
10 m/dev	$0.0483_{\pm 0.0720}$	$0.0661_{\pm 0.0802}$	$0.0921_{\pm 0.1015}$

the probability distribution of the link volatility for each network. The ordinate axis represents the volatility value. It has been divided in ten intervals. The rank of each interval is 0.1. In the abscises axis the probability of having this volatility during the simulation time is represented.

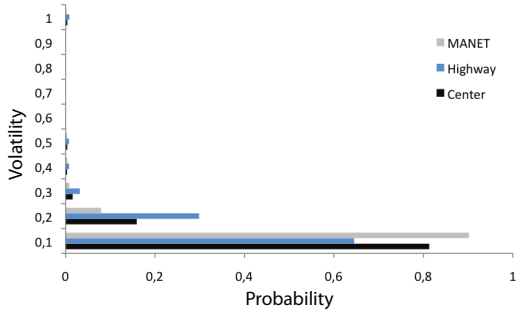
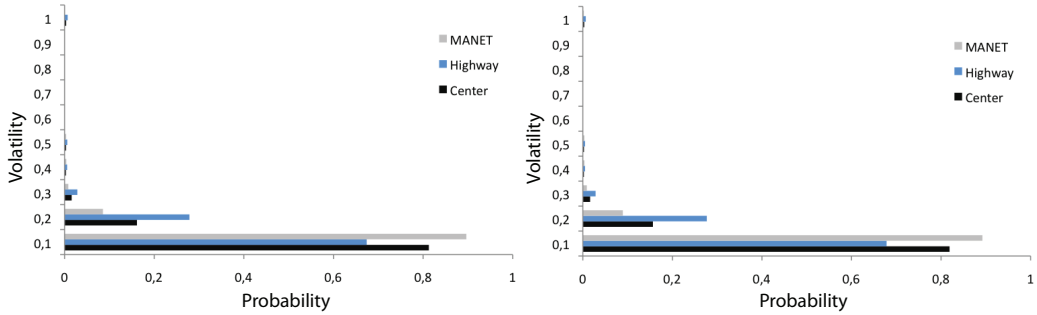


Figure 10: Probability distribution of the volatility in a density of 25 m per device



(a) 15 m per device

(b) 10 m per device

Figure 11: Probability distribution of the volatility in different densities.

As we can see in Figure 10 and 11, the same conclusion as in Table 4 arises: the volatility does not depend on the density of the network. The probability distribution obtained behaves similar in every density for each scenario. The probability values obtained for having a volatility lower than 0.1 are around: (1) 0.89 for the pedestrian area, (2) 0.81 for the vehicular city center, and (3) 0.67 for the highway. If we take into account that the maximum speed in the highway is 18 times bigger than the maximum speed in the MANET, we can consider that the volatility distribution shows that the highway is a stable topology.

7. Simulation Results

We are comparing the protocols in three scenarios: (1) pedestrian city center, (2) vehicular city center, and (3) highway; with three different network densities: (a) a dense network with 1 device per 10 meters, (b) a less dense with 1 device every 15 meters, and (3) the one with lowest density with 1 device per 25 meters. The protocols are compared using the simulator and the mobility model explained in Sect. 5. We are considering for this study a collision free network.

For comparing the five protocols, we measure the number of devices in the network that received the broadcasted message (hereinafter called broadcast coverage), and also the message complexity what means the number of broadcast and unicast needed to spread the message in the network.

In previous work, the first version of BODYF was compared also in terms of bandwidth used and broadcast time with DFCN and SF. That work concluded that BODYF is slower regarding the broadcast time, but the number of devices reached is much higher than for SF and DFCN. The values for the comparison regarding the time and the bandwidth can be found in [9]. Moreover, the cost of creating and maintaining the tree was also presented in this work in Sect. 6, Table 3.

The improved version of the implementation of DAGRS (thus BODYF) presented in this work implements more efficient operations in the communication level that the previous one using a smaller number of messages for the merging process. Experiments not included in this paper confirm that BODYF on the new implementation of DAGRS reached at least 6% more devices than the previous version reducing at the same time the number of forwarding nodes, that is the network usage. In this previous work, we studied BODYF with a different network configuration, i.e., the transmission

range of the devices was smaller, the densities studied were different and therefore, the network was very partitioned. We checked there that BODYF can achieve much better coverage than SF or DFCN, and that it was also able to spread the message outside the original partition. As the performance of BODYF was already tested in partitioned networks giving highly competitive results, in this work we wanted to check its performance when it is working with connected or nearly connected networks. So that in this situation, when the network is completely connected, SF will obtain the best coverage, but it will also be the upper bound for the network resources used. SF in a connected network will reach every single device in the system (not considering collisions), therefore, any protocol obtaining a similar coverage will be really satisfactory. For doing a better study, in this work we are also comparing with SAPF and WPB already described in Sect. 4.

As we mentioned before in Sect. 4, it is necessary to set some parameters for the correct functioning of the some of the protocols. For DFCN, after a tuning process, these threshold values were set as follows:

- $benefit = 0.4$.
- $RAD \in [0,7]$ seconds.
- The *densityThreshold* is different in each density.
  - a device each 25 m $\rightarrow$ $densityThreshold = 10$.
  - a device each 15 m $\rightarrow$ $densityThreshold = 12$.
  - a device each 10 m $\rightarrow$ $densityThreshold = 15$.

WPB also has some parameters that must be configured. Those are $WAIT\_TIME$ and δ. The former is the time set when a broadcasting message is received to allow duplicated messages to arrive to the same node. The latter is a timer that is established in case of suppressing in order to check if any neighbor sends the broadcast. In case the message of its neighbor is not heard, the node will forward it. Just as recommended in [13], both values are set to 5ms.

In our experiments, we disseminate a message every 30 seconds during a period of 10 minutes (that means we made 20 broadcasting processes starting from the same device, but from different positions since it is moving). This was simulated in 30 different topologies to make sure our results are reliable.

The results presented in this work are the average values obtained after these simulations.

The percentage of devices reached by every protocol is also shown in Table 5, where it can be seen that BODYF nearly covers the whole network in the two cases dealing with vehicles using less than 32% of forwarding nodes to achieve such a good coverage. SAPF shows a very bad performance for both MANET and Highway environments. On one hand, for the former as devices move very slow, the probability of forwarding does not varies and it is always 0.05, which is a very low value to disseminate the message as it can be seen, since it just covers 6.54% of devices at maximum. On the other hand, when dealing with the highway environment the number of forwarding nodes is highly increased as all devices whose speed is higher that 100 Km/h always resend the message (behaves as SF). But for the vehicular city center the performance shown is very competitive. WPB shows a good behavior for both the coverage achieved and the network resources, similar to the one obtained with BODYF. DFCN is always below any protocol for both parameters measured.

Table 5: Percentage of devices reached.

% Dev. Reached	Densities	MANET	Highway	Vehicular Center
BODYF	**25 m/dev**	50.68	94.52	99.50
	15 m/dev	80.84	98.97	99.87
	10 m/dev	94.41	99.71	100
DFCN	**25 m/dev**	45.85	73.55	78.85
	15 m/dev	71.08	86.18	90.42
	10 m/dev	87.16	95.26	92.26
SF	**25 m/dev**	89.87	100	100
	15 m/dev	99.30	100	100
	10 m/dev	99.99	100	100
SAPF	**25 m/dev**	5.59	97.98	96.85
	15 m/dev	6.14	99.11	99.82
	10 m/dev	6.54	99.28	100
WPB	**25 m/dev**	55.04	92.50	98.16
	15 m/dev	83.59	98.51	99.64
	10 m/dev	95.15	99.61	99.81

In Table 6, we show the results obtained by the broadcasting protocols in terms of the message complexity, i.e., the number of messages sent by the protocols (network use). As these five protocols just use the local information to decide whether to resend the message or not, and the message is forwarded using one simple broadcasting message, the complexity of the message can be also seen as the number of forwarding nodes. It is possible to check how BODYF for the all the environments is forwarding less messages when the density increased, but at the same time the coverage achieved is also

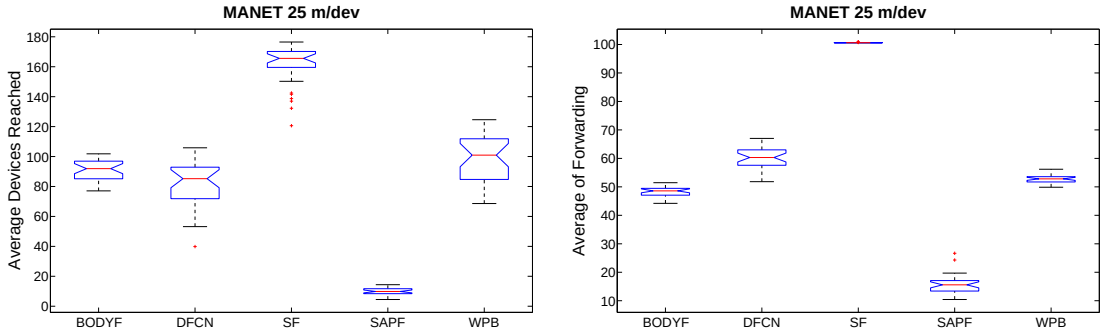
increasing. This is a very good performance since, as the network grows, the use of network resources decreased for achieving better coverage. The values shown in this table are the percentage of forwarding nodes that after receiving the message decides to rebroadcast it. Explaining why in DFCN, in the sparsest network, the percentage of rebroadcast is 60.17% while the coverage achieved is 45.85% (more than sixty percent of the 45.85% nodes receiving the message resend). As it can be seen, the worst protocol in terms of this ratio is SF, which always sends one message per device. SAPF presents a low number of nodes rebroadcasting in the pedestrian environment but it is due to the small reachability obtained. WPB, BODYF and DFCN show a similar performance for both vehicular scenarios. This is a very important result since DFCN that was specifically designed to reduce the number of forwarding messages, and needs a long process to be tuned for being competitive, meanwhile BODYF or WPB have no tune process.

Table 6: Percentage of forwarding nodes.

% Dev. Forwarding	Densities	MANET	Highway	Vehicular Center
BODYF	**25 m/dev**	48.20	31.57	30.04
	15 m/dev	37.95	27.75	25.88
	10 m/dev	34.73	24.28	22.04
DFCN	**25 m/dev**	60.17	26.73	27.81
	15 m/dev	56.24	24.59	26.59
	10 m/dev	52.41	23.23	26.63
SF	**25 m/dev**	100	100	100
	15 m/dev	100	100	100
	10 m/dev	100	100	100
SAPF	**25 m/dev**	15.08	76.22	30.20
	15 m/dev	10.72	75.87	30.01
	10 m/dev	8.62	75.88	29.94
WPB	**25 m/dev**	52.67	27.89	23.82
	15 m/dev	40.66	22.47	21.13
	10 m/dev	33.93	20.19	19.43

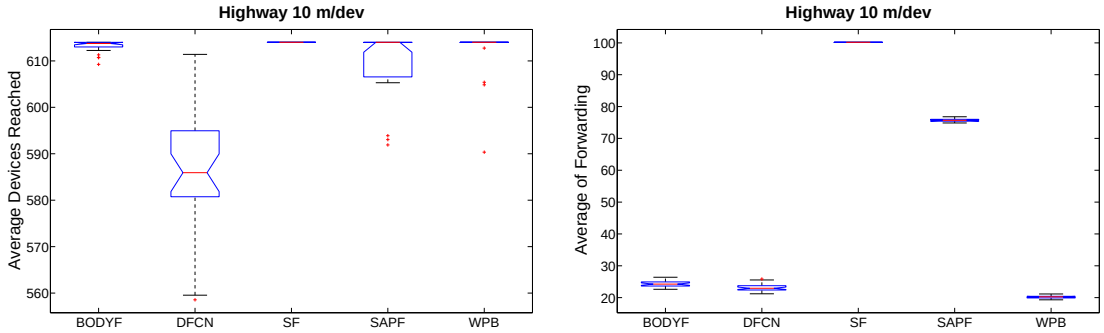
t

Statistical analysis have been made to all the results presented in this paper. This study has been done using the *boxplot* function from *Matlab*. In the displayed *boxplots*, the bottom and top of the boxes represent the lower and upper quartiles of the data distribution, respectively, while the line between them is the median. The whiskers are the lowest datum still within 1.5 IQR of the lower quartile, and the highest datum still within 1.5 IQR of the upper quartile. The crosses are data not included between the whiskers. Finally, the notches in the boxes display the variability of the median between samples. If the notches of two boxes are not overlapped, then it means that there is statistical significant difference in the data with



(a) Average of devices reached (b) Average of forwarding nodes

Figure 12: Statistical test for the MANET environment with 25 m/device.



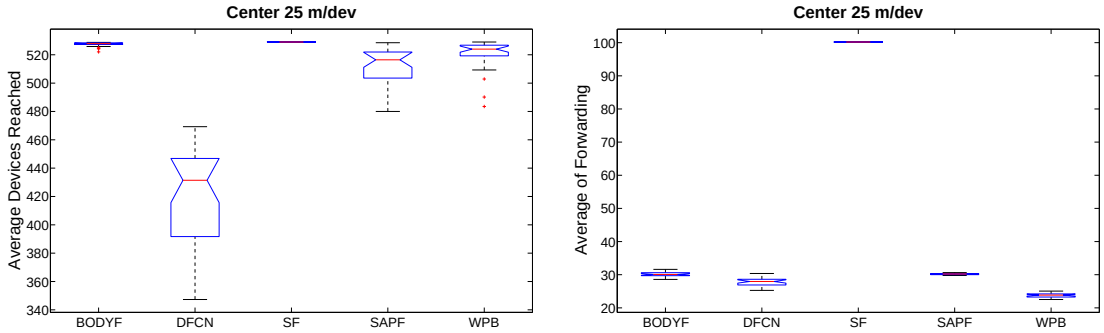
(a) Average of devices reached (b) Average of forwarding nodes

Figure 13: Statistical test for the Highway environment with10 m/device.

95% confidence.

Figure 12a shows that even when in average WPB obtained a higher number of devices reached than BODYF (see Table 5), the statistical analysis indicates that there are no significant differences between them, neither with DFCN. SF is the one with the best coverage achieved with statistical differences over all the rest. Meanwhile in Figure 12b, SAPF presents the best behavior over all the other protocols, followed by BODYF that also has significant differences with WPB, DFCN and SF.

For the highway environment shown in Figure 13 we can see in 13a that, there are no statistical differences between SF, BODYF, SAPF or WPB regarding the coverage achieved. In terms of the number of nodes forwarding,



(a) Average of devices reached

(b) Average of forwarding nodes

Figure 14: Statistical test for the vehicular city center environment with 25 m/device

in Figure 13b WPB shows the best behavior with statistical significance.

In Figure 14 the results for the sparsest density in the vehicular city center environment are presented. On one hand, it is possible to see how in Figure 14a, BODYF and SF has no statistical differences, but they two have with all the rest of protocols in terms of the number of devices reached. On the other hand, in terms of the average percentage of forwarding performs for the same network, Figure 14b shows that WPB behaves better than the rest with statistical difference.

Because of the lack of space, we did not include more statistical *boxplots*, but we will comment the more remarkable results obtained. For the number of devices reached, DFCN presents the worst behavior with significance differences in all the scenarios, except the MANET, were SAPF does not work at all and obtained worst results statistically. Contrary, as it was expected due to the high connectivity of the network, SF shows the best behavior. Regarding the number of forwardings performed, it is clear that SF is statistically the worst algorithm in all cases, followed by SAPF in both VANETs environment, but due to the bad performance in MANETs, the number of forwardings is also low, and therefore, statistically is the one using less network resources in the MANET scenario.

We did a ranking between each proposal in every density and scenario numbering each algorithm from 1 (the best) to 5 (the worst) according to the results obtained in the statistical tests. If there is not significance differences between two algorithms they are ranked in the same position. Summing up

Table 7: Ranking of algorithms.

	Coverage	Forwarding
BODYF	2	2
WPB	3	1
SF	1	5
DFCN	5	3
SAPF	4	4

the values obtained we get an overall ranking for the proposal that is shown in Table 7. The proposal obtaining the best position in the ranking for the coverage achieved is SF as it was expected because it is an upper bound (network very connected), but the second one was BODYF. In terms of the network resources or number of forwardings performed, WPB has the first position, but again BODYF is ranked the second one. So overall, BODYF shows a stable and reasonable behavior for all the scenarios studied.

8. Conclusions

In this paper, we present an improved version of the implementation of DAGRS that leads to a better performance of BODYF, a broadcasting protocol over a tree-based topology specifically designed for highly fluctuant topologies that does not have any parameter, and we compare its performance versus DFCN, SAPF, WPB and SF. DFCN is a neighbor based topology protocol, designed to minimize the resources required, and generally accepted by the community. SAPF is dissemination protocol that dynamically adapts the rebroadcasting probability in terms of the speed of the device. WPB is a distance based protocol that considers forwarding the message in terms of the distance between the source and the receiver nodes, And finally, Simple Flooding is one of the bases of broadcasting protocols, that simply foods the network. These protocols are compared in three very realistic environments, dealing with MANETs and VANETs, and using three different densities in each scenario. Thus, the protocols are finally compared in 9 different environments.

As BODYF works over a tree topology, we first check that the implementation of a tree topology in a vehicular ad hoc network is not affected by the high speed of the devices and hence, it is reasonable to use it. For measuring the performance of the tree topology, we consider again the 9 different test cases. The results show that the resources needed for creating a spanning

tree in a vehicular ad hoc network are not far from the ones a MANET requires, and therefore, it is feasible to apply this topology over a vehicular network.

The five different protocols have been compared in the 9 different scenarios mentioned before. Our results show that the coverage achieved by BODYF in general higher or does not have statistical differences with the others, excepts SF that is considered the upper bound. This is a very good performance since as networks are connected SF achieves the best possible coverage. Additionally, BODYF makes reasonable use of network resources, considering that for example DFCN was specifically created for reducing the number of forwarding messages.

BODYF was specifically designed for dealing with highly fluctuant networks, and the results show that the coverage it achieves when dealing with VANETs is really good making at the same time very efficient use of the network resources. Even when BODYF was not developed for MANETs, our experiments state that it is outperforming DFCN which was specifically tuned to work over MANETs, and also makes less use of the network resources that DFCN.

WPB also shows a very good performance in terms of the network resources, situated in the first position of the ranking done. It also achieves a very competitive coverage in all the scenarios.

As future work, we plan to compare BODYF to other source-tree based and cluster-based protocols to really know how good its behavior is versus other kind of approaches. We would also like to develop some applications on top of our protocol in such highly mobile networks, as it could be to propagate a large document, even downloading parts of this information from different devices if it is needed.

References

[1] Calm, continuous communication for vehicles, http://www.calm.hu/.

[2] Carlink project website, http://carlink.lcc.uma.es.

[3] E. Alba, B. Dorronsoro, Cellular Genetic Algorithms, Operations Research/Compuer Science Interfaces, Springer-Verlag Heidelberg, 2008.

[4] J. M. McQuillan, I. Richer, E. C. Rosen, The new routing algorithm for

the arpanet, IEEE Transactions on Communications COM-28 (5) (1980) 711–719.

[5] IEEE standard 802.1d-2004: IEEE Standard for Local and Metropolitan Area Networks: Media Access Control (mac) Bridges (June 2004).

[6] W. Zhao, M. Ammar, E. Zegura, Multicasting in delay tolerant networks: semantic models and routing algorithms, in: WDTN '05: Proceedings of the 2005 ACM SIGCOMM workshop on Delay-tolerant networking, ACM, New York, NY, USA, 2005, pp. 268–275.

[7] A. Piyatumrong, P. Bouvry, F. Guinand, K. Lavangnananda, Trusted spanning trees for delay tolerant mobile ad hoc networks, in: IEEE Conference on Soft Computing in Industrial Applications, 2008. SMCia '08, 2008, pp. 131–136.

[8] J. Su, A. Goel, E. de Lara, An empirical evaluation of the student-net delay tolerant network, Mobile and Ubiquitous Systems, Annual International Conference on (2006) 1–10.

[9] P. Ruiz, B. Dorronsoro, D. Khadraoui, P. Bouvry, L. Tardón, BODYF– A Parameterless Broadcasting Protocol Over Dynamic Forest, in: Proceedings of the High Performance Computing & Simulation Conference (HPCS 08), Nicosia, Cyprus, 2008, pp. 297–303.

[10] R. Handorean, C. D. Gill, G.-C. Roman, Accommodating transient connectivity in ad hoc and mobile settings, in: Pervasive, 2004, pp. 305–322.

[11] E. Alba, B. Dorronsoro, F. Luna, A. J. Nebro, P. Bouvry, L. Hogie, A cellular multi-objective genetic algorithm for optimal broadcasting strategy in metropolitan manets, Computer Communications 30 (4) (2007) 685–697.

[12] Y. Mylonas, M. Lestas, A. Pitsillides, Speed adaptive probabilistic flooding in cooperative emergency warning, in: WICON '08: Proceedings of the 4th Annual International Conference on Wireless Internet, 2008, pp. 1–7.

[13] N. Wisitpongphan, O. Tonguz, J. Parikh, P. Mudalige, F. Bai, V. Sadekar, Broadcast storm mitigation techniques in vehicular ad hoc networks, Wireless Communications, IEEE 14 (6) (2007) 84 –94.

[14] B. Williams, T. Camp, Comparison of broadcasting techniques for mobile ad hoc networks, in: Proceedings of the ACM International Symposium on Mobile Ad Hoc Networking and Computing (MOBIHOC), 2002, pp. 194–205.

[15] D. Li, H. Huang, X. Li, M. Li, F. Tang, A distance-based directional broadcast protocol for urban vehicular ad hoc network, in: Wireless Communications, Networking and Mobile Computing, 2007. WiCom. International Conference on, 2007, pp. 1520 –1523.

[16] O. Tonguz, N. Wisitpongphan, F. Bai, P. Mudalige, V. Sadekar, Broadcasting in vanet, in: Mobile Networking for Vehicular Environments, 2007, pp. 7 –12.

[17] O. Tonguz, N. Wisitpongphan, F. Bai, Dv-cast: A distributed vehicular broadcast protocol for vehicular ad hoc networks, Wireless Communications, IEEE 17 (2) (2010) 47 –57.

[18] S.-Y. Ni, Y.-C. Tseng, Y.-S. Chen, J.-P. Sheu, The broadcast storm problem in a mobile ad hoc network, in: 5th annual ACM/IEEE international conference on Mobile computing and networking, 1999, pp. 151–162.

[19] S. Bai, Z. Huang, D. Kwak, S. Lee, H. Oh, J. Jung, Vehicular multihop broadcasting protocol for safety message dissemination in vanets, in: Vehicular Technology Conference Fall (VTC 2009-Fall), IEEE 70th, Anchorage, AK, 2009, pp. 1 –5.

[20] L. Hogie, P. Bouvry, F. Guinand, G. Danoy, E. Alba, A Bandwidth-Efficient Broadcasting Protocol for Mobile Multi-hop Ad hoc Networks, in: Demo proceeding of the 5th International Conference on Networking (ICN'06), IEEE, 2006, p. 71.

[21] L. Zhou, G. Cui, H. Liu, Z.Wu, D. Luo, Nppb: A broadcast scheme in dense vanets, Information Technology Journal 9 (2) (2010) 247 –256.

[22] M. Slavik, I. Mahgoub, Stochastic broadcast for vanet, in: Consumer Communications and Networking Conference (CCNC), 7th IEEE, Las Vegas, 2010, pp. 1 –5.

[23] C.-M. Huang, L. Tu, C.-H. Chou, Rewarn: An opportunistic relay scheme for cooperative collision warning in vanet, in: Personal, Indoor and Mobile Radio Communications, IEEE 20th International Symposium on, Tokyo, 2009, pp. 3030 –3034.

[24] J. Schleich, G. Danoy, P. Bouvry, A. Le Thi Hoai, Blackbone2, an efficient deterministic algorithm for creating 2-connected m-dominating set-based backbones in ad hoc networks, in: Proceedings of the Seventh ACM International Workshop on Mobility Management & Wireless Access, Tenerife, 2009, pp. 91–98.

[25] A. Casteigts, Model driven capabilities of the DA-GRS model, in: Intl. Conf. on Autonomic and Autonomous Systems (ICAS'06). San Francisco. USA, 2006, p. 24.

[26] A. Piyatumrong, P. Ruiz, P. Bouvry, F. Guinand, K. Lavangnananda, Traversal strategies of a distributed spanning forest algorithm in mobile ad hoc - delay tolerant networks, in: In Proceedings of the 3rd International Conference on Advances in Information Technology (IAIT'09), Bangkok, Thailand, 2009, pp. 96–109.

[27] L. Hogie, Mobile ad hoc networks, modelling, simulation and broadcast-based applications, Ph.D. thesis, University of Luxembourg (2007).

[28] A. Andronache, S. Rothkugel, Hytrace—backbone-assisted path discovery in hybrid networks, in: The Seventh International Conference on Networking ICN, 2008, pp. 34–40.

[29] D. Gorgen, H. Frey, C. Hiedels, JANE–the Java Ad-hoc Network Environment, in: Proceedings of the 40th Annual Simulation Symposium, 2007, pp. 163–176.

[30] Y. Pigné, Modèlisation et traitement décentralisé des graphes dynamiques – application aux réseaux mobiles ad hoc, Ph.D. thesis, Université du Havre (2008).