Journal of Universal Computer Science, vol. 13, no. 1 (2007), 4-31
submitted: 29/8/05, accepted: 23/8/06, appeared: 28/1/07 © J.UCS

Duplicate Address Detection and Autoconfiguration in OLSR

Saadi Boudjit[1], **Cédric Adjih**[2], **Paul Mühlethaler**[2], **Anis Laouiti**[3]
[1] GET/ENST, 37/39, rue Dareau 75014 Paris, France.
[2] INRIA, BP 105, Rocquencourt, 78153 Le Chesnay Cedex, France.
[3] GET/INT, 9, rue Charles Fourier, F-91011, France.
Saadi.Boudjit@enst.fr, Cedric.Adjih@inria.fr, Paul.Muhlethaler@inria.fr,
Anis.Laouiti@int-evry.fr

Abstract: Mobile Ad hoc NETworks (MANETs) are infrastructure-free, highly dynamic wireless networks, where central administration or configuration by the user is very difficult. In hardwired networks nodes usually rely on a centralized server and use a dynamic host configuration protocol, like DHCP [Droms et al. 2003], to acquire an IP address. Such a solution cannot be deployed in MANETs due to the unavailability of any centralized DHCP server. For small scale MANETs, it may be possible to allocate free IP addresses manually. However, the procedure becomes impractical for a large-scale or open system where mobile nodes are free to join and leave.
Most of the autoconfiguration algorithms proposed for ad hoc networks are independent of the routing protocols and therefore, generate a significant overhead. Using the genuine optimization of the underlying routing protocol can significantly reduce the autoconfiguration overhead. One of the MANET protocols which have been promoted to experimental RFC is the *OLSR* routing protocol [Jacquet et al. 2003], on which this article focuses. This article aims at complementing the *OLSR* routing protocol specifications to handle autoconfiguration. The corner stone of this autoconfiguration protocol is an advanced duplicate address detection algorithm.
Key Words: MANET, Autoconfiguration, OLSR
Category: C.2.0, C.2.1, C.2.2

1 Introduction

Many fruitful efforts have focused on routing protocols for *MANET* in recent years, dealing essentially with the routing issue itself, and not necessarily considering the autoconfiguration functionality. These MANET protocols can be classified into proactive protocols [Jacquet et al. 2003] where each node maintains an up-to-date version of the network topology by periodic exchange of control messages with neighboring nodes; and reactive protocols [Perkins et al. 2003] where each node discovers the route to a destination on demand.

Research on automatic configuration of IP addresses for *MANET* is relatively less frequent. The IPv6 and ZEROCONF working groups of the IETF deal with autoconfiguration issues but with a focus on wired networks. Automatic address allocation is more difficult in a *MANET* environment than in wired networks due to instability of links, mobility of the nodes, the open nature of the mobile ad hoc networks, and lack of central administration in the general case. Thus performing a DAD (Duplicate Address Detection) generates more complexity and more overhead in ad hoc networks than in wired networks where protocols such as DHCP [Droms et al. 2003] and SAA [Thomson and Narten 1998] can be used.

In this paper we will describe an autoconfiguration solution for the *OLSR* (Optimized Link State Routing) protocol. This solution is based on an efficient Duplicate Address Detection (DAD) algorithm which takes advantage of the genuine optimization of the *OLSR* protocol.

The paper is structured as follows: Section 2 is dedicated to related work, mostly concerning previously proposed autoconfiguration protocols in ad hoc networks. Section 3 gives the main features of the *OLSR* routing protocol. Then Section 4 describes the duplicate address detection mechanism which is the core of our proposed autoconfiguration protocol. A formal proof of correctness of this detection algorithm is given. Section 5 proposes different ways to assign an initial IP address to a newly arriving node and to resolve address duplication. The overhead generated by the autoconfiguration algorithm and some simulation results are also provided. This section actually completes the previous one on duplicate address detection. These two sections constitute a complete autoconfiguration algorithm. The paper concludes in Section 7.

2 Related work

Numerous studies have been carried out on autoconfiguration protocols and the related issue of duplicate address detection in ad hoc networks. These studies have been proposed within the IETF or published in academic papers. This section is organized as follows. First, we review autoconfiguration scenarios and the different conditions where address duplications may occur. Then, we briefly present the various proposed autoconfiguration protocols. To conclude this section we position our contribution with respect to the previously proposed autoconfiguration protocols.

2.1 Address autoconfiguration scenarios and address duplications

Before describing algorithms, we first highlight some scenarios where address duplications may occur and which allow to discriminate between the different algorithms.

The first scenario is the simplest: a mobile node joins and then leaves a *MANET*. An unused IP address is allocated to the node on its arrival and becomes free on its departure. In some scenarios, the allocated IP address may be duplicated in the network.

A more complicated scenario is the following: nodes are free to move arbitrarily in the *MANET* and, consequently, the network may becomes partitioned. In the resulting partitions, the nodes continue to use the previously allocated IP addresses. If a new node comes to one of the partitions, it may be assigned an IP address belonging to another partition. Address duplication may occur when the partitions merge later.

Another scenario is when two independent *MANETs* merge. Because the two *MANETs* were configured separately, and the address allocation in each of the *MANETs* is independent of the other, there may be duplicated addresses when the two *MANETs* merge.

Most of the other scenarios, can be thought of as special cases of the three scenarios described above.

2.2 Address autoconfiguration in ad hoc networks

There are two main approaches to address autoconfiguration in ad hoc networks. The first approach tends to allocate conflict free addresses. The algorithms in this approach are often called conflict-free allocation algorithms. The second approach tends to allocate addresses on a random basis and uses dedicated mechanisms to detect duplicate addresses. When duplications are detected, new addresses with new values are assigned.

The Distributed Dynamic Host Configuration Protocol (DDHCP) [Nesargi and Prakash 2002] is one example of a conflict-free allocation algorithm. In this algorithm, the nodes responsible for allocation try to assign an unused IP address to a new node – unused to the best of their knowledge. Then the new node performs DAD to guarantee that it is an unallocated IP address. DDHCP maintains a global allocation state, so IP addresses which have been used, and addresses which have not yet been allocated, are known. When a new node joins the network, one of its neighbors choses an unused address for it. The same unused IP address in the global address pool could be assigned to more than one node arriving at almost the same time. This is the reason why DAD is still performed by a node after getting an IP address. This algorithm takes into account network partition and merger, and works well with proactive routing protocols.

Dynamic Configuration and Distribution Protocol (DCDP) [Misra et al. 2001] is another conflict-free allocation algorithm. When a new mobile node joins the *MANET*, an address pool is divided into halves between itself and a configured node. This algorithm takes into account network partition and merge, however conflicts will occur during the merge if two or more of the separately configured *MANETs* taking part in the merge, begin with the same reserved address range.

Another conflict-free allocation algorithm, called the 'prophet allocation protocol' has been proposed for large scale *MANETs* in [Zhou et al. 2003]. The idea is that every mobile node executes a stateful function *f(n)* to get a unique IP address. *f(n)* is function of a state value called the *seed* which is updated for each node in the network. In this algorithm, mechanisms are proposed for network partition and merger. The difficulty in this solution is to find a function *f(n)* which will guarantee the generation of unique IP addresses each time the function is executed by a node.

The algorithms related to the second approach, perform a DAD (Duplicate Address Detection) to ensure the uniqueness of the allocated IP address. The general procedure is that a node generates a tentative address and then performs DAD within its neighborhood (radio range of the node). If the address is unique, the DAD is performed again over the whole network and a unique IP address is constructed. Examples of such approaches include [Boudjit et al. 2004], [Perkins et al. 2001] and [Weniger and Zitterbart 2002].

DAD mechanisms can also be divided into two categories which differ in when, and how duplicate addresses are detected.

The ADAD (Active Duplicate Address Detection) mechanisms distribute additional information in the network to prevent address duplication as, for instance, in [Perkins et al. 2001] and [Weniger and Zitterbart 2002].

In contrast, PDAD (Passive Duplicate Address Detection) algorithms [Weniger 2003], try to detect duplicates without disseminating additional control information in the network. The idea behind this approach is to continuously monitor routing protocol traffic to detect duplicates rather than sending additional control packets for this purpose. However, in [Weniger 2003] a so-called Address Conflict Notification (ACN) message is introduced for the purpose of conflict resolution, and no less than nine different approaches to detecting duplicated addresses have been presented for proactive link-state routing protocols. The advantage of this solution consists in introducing only one additional message used to report a conflict. This saves bandwidth in the absence of conflictual situations. However several drawbacks can be found. First it is a heavy solution because of the numerous algorithms that must be combined to handle all the scenarios. For instance for OLSR, due to the peculiar difficulties induced by the MPR optimization, more than nine different algorithms are necessary to cover all the identified scenarios. At the same time to avoid possible storm effects when a duplicate address is detected, special techniques must be used to control the flooding of ACN messages. Finally there is no formal proof that all duplications can be finally detected.

2.3 Our contribution

In this paper we are proposing a new autoconfiguration algorithm for OLSR based on a DAD approach. The main idea is that preventing address collision requires an assumption that some unique identifiers (*Node-ID*) are allocated to each node in the network. As for the PACMAN [Weniger 2003] proposal, this new autoconfiguration algorithm is optimized to reduce the bandwidth utilization. As a matter of fact this approach uses the OLSR's MPR optimization to broadcast a new control packet(*MAD: Multiple Address Decalaration*) used by the DAD procedure. Moreover this approach is extremely simple and a formal proof of correctness is given in the article. We are now going to give a short review of the *OLSR* routing protocol.

3 OLSR and MPR technique

This section describes the main features of the *OLSR* (Optimized Link State Routing) protocol. *OLSR* is an optimization of a pure link state routing protocol. It is based on the concept of *multipoint relays* (*MPRs*) [Qayyum et al. 2002]. First, using *multipoint relays* reduces the size of the control messages: rather than declaring all links, a node declares only the set of links with its neighbors that are its "*multipoint relay selectors*". The use of *MPRs* also minimizes the flooding of control traffic. Indeed only *multipoint relays* forward control messages (Figure 1). This technique significantly reduces the number of retransmissions of broadcast control messages [Jacquet et al. 2003] [Qayyum et al. 2002]. The two main *OLSR* functionalities, Neighbor Discovery and Topology Dissemination, are now detailed. Then we present *OLSR* "gateway" mechanism used by some nodes to declare reachability to their connected hosts and networks.

3.1 Neighbor Discovery

Each node must detect the neighbor nodes with which it has a direct link.

For this, each node periodically broadcasts *Hello* messages, containing the list of neighbors known to the node and their link status. The link status can be either *symmetric* (if communication is possible in both directions), *asymmetric* (if communication is only possible in one direction), *multipoint relay* (if the link is symmetric and the sender of the *Hello* message has selected this node as a *multipoint relay*), or *lost* (if the link has been lost). The *Hello* messages are received by all 1-hop neighbors, but are not forwarded. They are broadcasted once per refreshing period called the "*HELLO_INTERVAL*" (the default value is 2 seconds). Thus, *Hello* messages enable each node to discover its 1-hop neighbors, as well as its 2-hop neighbors. This neighborhood and 2-hop neighborhood information has an associated holding time, the - "*NEIGHBOR_HOLD_TIME*",

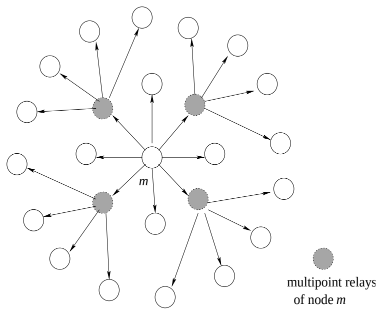


Figure 1: Multipoint relays of node m

after which it is no longer valid.

On the basis of this information, Each node m independently selects its own set of *multipoint relays* among its 1-hop neighbors in such a way all 2-hop neighbors of m have *symmetric* links with $MPR(m)$. This means that the *multipoint relays* cover (in terms of radio range) all 2-hop neighbors (Figure 1). One possible algorithm for selecting these *MPRs* is described in [Qayyum et al. 2002]. The *MPR* set is computed whenever a change in the 1-hop or 2-hop neighborhood is detected. All details of *OLSR* protocol can be found in the IETF RFC 3626 document [Jacquet et al. 2003]. In addition, each node m maintains its "*MPR selector set*". This set contains the nodes which have selected m as a *multipoint relay.* Node m only forwards broadcast messages received from one of its *MPR selectors.*

3.2 Topology Dissemination

Each node of the network maintains topological information about the network obtained by means of *TC (Topology control)* messages. Each node m selected as a *multipoint relay*, broadcasts a *TC* message at least every "*TC_INTERVAL*" (the default value is 6 seconds). The *TC* message originated from node m declares the *MPR selectors* of m. If a change occurs in the *MPR selector* set, the next *TC* can be sent earlier. (e.g. after some pre-specified minimum interval). The *TC* messages are flooded to all nodes in the network and take advantage of *MPRs* to reduce the number of retransmissions. Thus, a node is reachable either directly or via its *MPRs.* This topological information collected in each node has an associated holding time "*TOP_HOLD_TIME*", after which it is no longer valid.

The neighbor information and the topology information are refreshed periodically, and they enable each node to compute the routes to all known destinations. These routes are computed with Dijkstra's shortest path algorithm [Tanenbaum 1996]. Hence, they are optimal as concerns the number of hops. Moreover, for any route, any intermediate node on this route is a *multipoint relay* of the next node. The routing table is computed whenever there is a change in neighborhood or topology information.The flooding through the MPR and the fact that only *MPR selectors* are announced in TC allow one to keep the overhead incurred by TC message at low level even in large and dense networks.

OLSR can use other control messages to announce multiple interfaces or gateway. These messages are not used in the following and thus are not not described here.

4 Duplicate Address Detection

Our proposed autoconfiguration algorithm is based on two steps. In the first step, an IP address is selected by the arriving node and this latter can join the ad hoc network. Numerous schemes can be used to select this IP address. For instance the node can perform a random selection in a well known pool of addresses; another technique consists of one of the neighbors selecting the address on behalf of the arriving node. In Section 5 we discuss in detail various ways to assign an IP address to an arriving node.

After this first step has been performed, the second step can take place. The aim of this step is to detect potential address duplications on run. To perform this task a DAD(Duplicate Address Detection) algorithm is started on this newly configured node. This DAD algorithm allows the newly configured node to state whether the selected address is duplicated or not in a proactive manner. If such a case occurs, a node can change its address with respect to some specified criteria.

In order to detect address conflicts, each node diffuses a special message that we call MAD for "Multiple Address Declaration" to the entire network. This control packet includes the node address and the node identifier. The node identifier is a sequence of bits of fixed length L which is randomly generated. Hence we are using the standard idea that the probability of two nodes having the same node identifier is low, and the probability of at least one address collision with N nodes, which is the well known "birthday problem" [Sayrafiezadeh 1994], can be set arbitrarily low by choosing a large enough value of L.

A node detects that it is in conflict when it receives a MAD message having the same address as its own, but with a different identifier. Actually other nodes will detect the conflict by receiving two MAD messages having the same address, but holding different node identifiers as is the case for node C in Figure 2. These

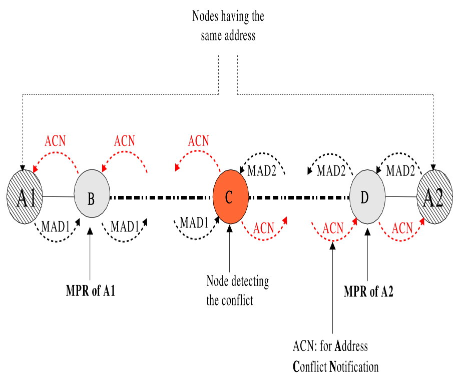

Nodes having the same address
ACN
ACN
ACN
MAD2
MAD2
MAD2
A1
B
C
D
A2
MAD1
MAD1
MAD1
ACN
ACN
ACN
MPR of A1
Node detecting the conflict
MPR of A2
ACN: for Address Conflict Notification

Figure 2: Conflict Notification

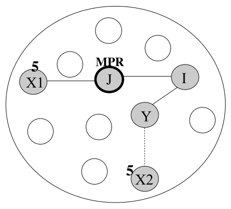


Figure 3: Address duplicate scenario

nodes could announce the conflict using a special control message as is done in the PACMAN protocol [Weniger 2003]. However this approach may induce broadcast storm since many nodes may announce the conflict and special care must be taken to avoid this effect. For that reason we do not recommend this way. An efficient manner to notify the address duplication to the nodes in conflict, consists in the MAD packets being received by all the nodes in the network. To save the channel bandwidth the MAD packets should be broadcasted using the MPR-flooding. Actually, applying OLSR relaying optimization rules as they are defined, may not be sufficient to ensure diffusion in some conflictual cases. As an illustration of such possible situations, we give the following example.

Figure 3 shows two conflicting nodes $X1$ and $X2$ ($X1$ and $X2$ have the same address 5), in the 2-hop neighbors of node I. In this configuration, nodes Y and I are not chosen as MPRs, then, the "Multiple Address Declaration" messages diffused respectively by the nodes $X1$ and $X2$ can not be propagated throughout the entire network. In our scenario, node I could not calculate its MPR set correctly, because MPR calculation is based on the assumption that there is no address duplication in the 1-hop and 2-hop neighbors. Consequently, node $X1$ and node $X2$ will not detect the address conflict, and the network remains corrupted. To handle such cases, new rules are added to the classical MPR-flooding algorithm for MAD message diffusion. This modified version of MPR-flooding takes into consideration the possibility that MPR originator addresses might be duplicated.

The new version of MPR-flooding algorithm is only useful for “Multiple Address Declaration”, but could be used in general for any kind of message which includes the node identifier. This version of MPR-flooding is called *Duplicate Address Detecting MPR Flooding* or *DAD-MPR Flooding*.

4.1 Special rules of the DAD-MPR flooding and proof of correctness

DAD-MPR flooding should have the property that it will continue to work even in the presence of duplicate addresses in the network. This property means that, in the absence of packet loss, the DAD-MPR Flooding will allow a MAD packet to reach all the nodes in the network. First, we will have the weaker property stating that for two nodes A_1 and A_2 using the same address A, but holding different node identifiers ID_{A_1} and ID_{A_2} the MAD message of A_1 will be received by A_2 and vice versa the MAD message of A_2 will be received by A_1.

To define and prove the correctness of the DAD-MPR flooding algorithm assuring the previous property, we will make the following assumptions. We will assume that there are only two nodes with a duplicated address in the network. In the following, we will see that this assumption can actually be significantly weakened. Moreover we will assume that there is permanent packet loss between a node and its neighbors. Thus a packet broadcast by a node will eventually reach all its neighbors. This assumption is true in a connected MANET when the packet is sent periodically and the network load is kept below a reasonable level.

Let us consider one pair of nodes which have duplicate addresses. Let us denote by A_1 and A_2 these two nodes with same address A, but different node identifiers ID_{A_1} and ID_{A_2} and let us denote by d the distance, in number of hops, between A_1 and A_2. Let us assume that both of them send a MAD message M_1 and M_2.

We intend to add special rules to the classical MPR-flooding algorithm to handle the MAD messages diffusion. With these additional rules we must be in a position to prove that if there is no packet loss the MAD message of A_1 will be received by A_2 and vice versa the MAD message of A_2 will be received by A_1. The main problem here comes from the MPR calculation as we have

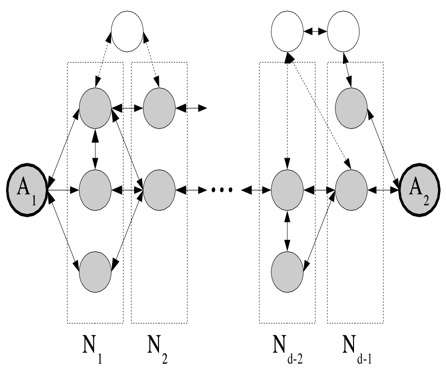


Figure 4: Example of Sets N_i

already seen in a previous example. As a matter of fact, the MPR calculation is done on addresses so the MPR set of a node may not be properly calculated if it has address duplication in its 1-hop and 2-hop neighborhood. Therefore, the MPR set may not properly cover the 2-hop neighborhood of this node. In such a case, control messages using the MPR optimization may not be propagated to all nodes in the network.

Let us denote by N_i the set of nodes which are exactly at distance i of A_1 and at distance $d-i$ of A_2, for $i \in \{1, ..., d-1\}$. Those nodes are precisely the nodes which are on a shortest path from A_1 to A_2. Hence the sets N_i are never empty since a shortest path exists. An example of such sets is illustrated in Figure 4.

Then several cases can occur, depending on the distance d:

4.1.1 $d \geq 5$

In Figure 5, nodes A_1 and A_2 have the same address, and they are 5 hops away from each other. In this case nodes A_1 and A_2 and all the intermediary nodes do not have duplications in their 1-hop and 2-hop neighbors according to our assumption of only two nodes with a duplicated address. Hence all the intermediary nodes calculate their MPR set properly. So, using the MPR-flooding the messages M_1 and M_2 will be correctly propagated to the nodes A_2 and A_1 respectively and the conflict will be detected. With the same consideration we can show that, in this case, all the MAD messages will reach all the nodes in the network.

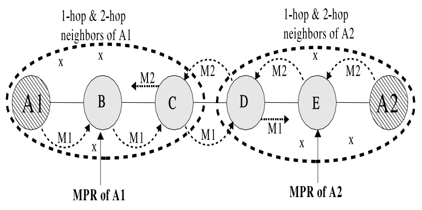


Figure 5: $Distance \geq 5$

4.1.2 $d = 4$

In Figure 6, nodes A_1 and A_2 do not have duplications in their 1-hop and 2-hop neighbors according to our initial assumption. Therefore, the MAD messages diffused by A_1 and A_2 will reach node C. Node C can detect the conflict by examining the node identifiers contained in the received MAD messages. These messages should be relayed by C, because it has been chosen as a MPR by B and D. In our case, this is not trivial. In fact, messages generated by A_1 and A_2 may have close sequence numbers, which may prevent one of the two sets of messages from being relayed by C (due to possible existence of a duplicate tuple indicating that such a message having the same sequence number and originator, has been received and processed). We need here to add an extra rule to allow MAD message relaying. We modify the MAD duplicate message detection, which will be based on the node originator address, the message sequence number, plus the node identifier. Hence, A_1 and A_2 MAD messages will be forwarded by C in all cases and reach B and D. Notice here that the MPR calculation of C is affected due to the presence of duplicated addresses in its 2-hop neighbors (A_1 and A_2). C chooses only one node between B and D as a MPR to cover its 2-hop neighbors. The chosen MPR, should act like node C as described before to relay the MAD messages. Following this rule, we are sure that one of the two nodes (A_1 and A_2) receives the MAD messages generated by the other node and hence can detect the conflict. We call this rule *Rule 1.*
We will see with the next case, where $d = 3$, another rule that allows the two conflicting nodes to detect the address duplication.

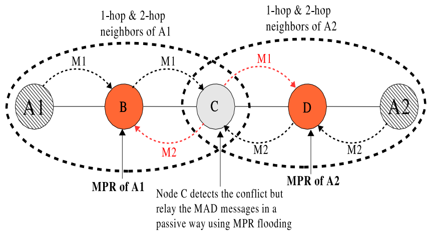

1-hop & 2-hop neighbors of A1
1-hop & 2-hop neighbors of A2
M1
M1
M1
A1
B
C
D
A2
M2
M2
M2
MPR of A1
MPR of A2
Node C detects the conflict but relay the MAD messages in a passive way using MPR flooding

Figure 6: Distance = 4

4.1.3 $d = 3$

In case of distance $d = 3$, nodes B and C (Figure 7) do not need to choose a MPR to cover respectively their 2-hop neighbors A_2 and A_1 since they have the same address. Address A is considered as a one hop neighbor. In contrast, A_1 chooses node B as a MPR to reach node C, and node A_2 chooses C as a MPR to reach node B. In this situation, and thanks to the new rule described previously, the A_1 MAD messages will reach node C, and the ones generated by A_2 will arrive at node B. But, B and C do not choose each other as a MPR, consequently, A_1 can not receive MAD messages coming from A_2, and A_2 MAD messages will never reach A_1 node. To tackle this problem, we add another rule that we call ,*Rule 2*, to enable MAD relaying for such situations, as follows: if a given node N receives a MAD message from a neighbor M, and M did not select N as a MPR, then, node N will repeat this message if it detects that one of its 1-hop neighbors has the same address as the one contained in the MAD message. The MAD TTL value is put to 1 to avoid the transmission of the MAD message beyond the conflicting nodes.

4.1.4 $d = 2$

In Figure 8, the node B detects the duplication because the nodes $A1$ and $A2$ are in its 1-hop neighborhood, it proceeds by the same manner as in the case of $d = 3$. Thus node A_1 will receive the MAD message of A_2; and node A_2 will receive the MAD message of A_1.

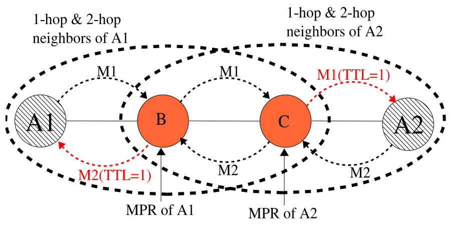


Figure 7: Distance = 3

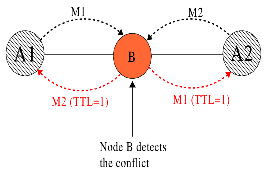


Figure 8: Distance = 2

4.1.5 $d = 1$

This is the simplest case and because nodes A_1 and A_2 are in the radio range of each other, the conflict will be detected by both nodes (Figure 9).

In this analysis, we consider the case of a unique couple of nodes having the same address in the network. Notice that, if we have a single couple of nodes having the same address in a 2-hop neighborhood of each node in the network, the previous reasoning continues to work correctly. In fact, with the previous reasoning, any 2-hop conflict is detected and fixed. Hence, the MPR-flooding mechanism will work correctly. Consequently, MAD messages can be delivered to any other node in the network beyond the previously MPR corrupted area and finally other possible conflicts can be resolved.

The general case of multiple conflicts is treated in Section 4.1.6.

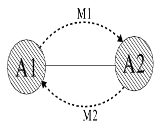


Figure 9: Distance = 1

4.1.6 Case of multiple conflicts

By multiple conflicts we mean, that we may have more than a single duplicated address in the network, knowing that a duplicated address, could be shared by several nodes at the same time. In the case of a duplicated address shared by more than two nodes in the network, conflicts are detected and fixed couple by couple by applying the previous rules cited in Section 4.1. Eventually, this kind of conflict is resolved. Nevertheless, the previous rules are not sufficient for the special case of loops as depicted in Figure 10. In fact, in Figure 10, each node considers that it has only two neighbors at 1-hop distance and no 2-hop neighbors (i.e the network seen by node A is composed by direct neighbors B and C). None of the nodes present in this network will be elected as a MPR. Hence, MAD messages will not be relayed and never reach other conflicting nodes or at least a neighbor of a conflicting node. In that case the previous rules will not ensure the relaying of MAD messages between nodes in conflict.

To handle multiple conflicts, we add a third rule to the classical MPR-flooding mechaninsm. The property that we add is actually simple. We weaken the relaying condition for nodes who are in the 1-hop neighborhood of a node who is sending an MAD message. When these neighbor nodes receive an MAD message, they must relay it irrespectively of the relaying conditions of the OLSR MPR-flooding algorithm (Figure 11). We call this rule *Rule 3.*

With these three rules, we will be in the position to prove the correctness of the DAD-MPR flooding algorithm. More precisely in the absence of packet loss an MAD message will finally reach all the nodes in the network.

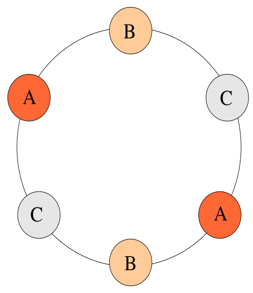
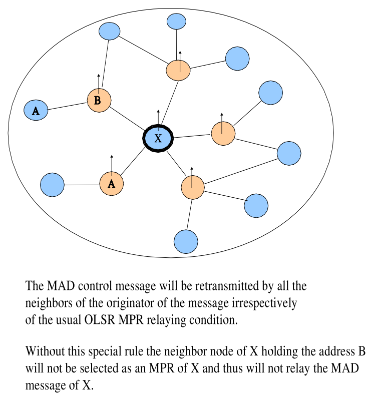


Figure 10: Case of multiple conflicts



The MAD control message will be retransmitted by all the neighbors of the originator of the message irrespectively of the usual OLSR MPR relaying condition.

Without this special rule the neighbor node of X holding the address B will not be selected as an MPR of X and thus will not relay the MAD message of X.

Figure 11: All the 1-hop neighbors of the originator of the MAD message will relay the message

Proof of correctness of DAD-MPR flooding algorithm with multiple conflicts

Let's denote: A, B, C, D, ... the nodes and '1', '2', ... the addresses. Let's denote 'A{1}' the node 'A' with the address '1' and so on ...
In this part, the conflicts with distance ≤ 2 are obviously resolved. The proof given in Section 4.1.3, can be adapted to the context of multiple conflicts by applying *Rule 2* and *Rule 3*. Thus any conflict at distance ≤ 3 is detected and then resolved.

Case of d = 4

Lemma 1 : in a permanent 4-hop conflict represented by the topology A{1}–

B{2}–C{3}–D{4}–E{1}, neither B nor D chooses C as MPR. In other terms, in a 4-hop conflict between two given nodes, the node(s) at the center of the conflict isn't chosen as MPR by the neighbors of the conflicting nodes. A node "in the center of conflict" is defined to be exactly 2-hop away from the both nodes in conflict. At least one such node exists by definition of "4-hop conflict".

Proof : by contradiction.

Assume there is a permanent 4-hop conflict: A{1}–B{2}–C{3}–D{4}–E{1} and C is MPR of B for instance (case with D is symetrical).

Then:

– The node A originates one MAD message

– The node B retransmits it: because it is a 1-hop neighbor of the MAD message originator (*Rule 3*)

– The node C retransmits it: because it is a MPR of B

– The node D retransmits it: because it detects the conflict and is one hop away from the other node in conflict (*Rule 2*).

which results in the conflict being resolved, which itself is contradictory with the "permanent 4-hop conflict" hypothesis.

Lemma 2 : in a permanent 4-hop conflict which can be represented by the topology: A{1}–B{2}–C{3}–D{4}–E{1}, there must be some nodes X and Y such as the topology includes: X{4}–Y–B{2}–C{3}–D{4}–E{1} and Y is MPR of B.

Proof :

Assume there is a permanent 4-hop conflict which can be represented by the topology: A{1}–B{2}–C{3}–D{4}–E{1}. *Lemma 1* shows that C is not MPR of B (nor D, incidentally). And then since D{4} is a 2-hop neighbor of B via C, proper MPR selection in B requires that:

(a) Some node with address 4 is covered by another 1-hop neighbor, chosen as MPR by B.

(b) OR some node with address 4 is a neighbor of B.

The last case (b) is impossible because ,otherwise, there would be a topology such as X{4}–B{2}–C{3}–D{4}–E{1}, which is a 3-hop conflict [1].
So (a) must be verified: let us denote X{4} the other 2-hop neighbor of B with address 4, and Y the 1-hop neighbor to reach it (the MPR chosen by B) [hence a part of the topology is represented by B{2}–Y–X{4}].

Then the topology includes:

[1] Recall that any conflict with distance ≤ 3 is resolved.

```
– B{2} – C{3} – D{4} – E{1}
   |
   Y
   |
  X{4}
```

Which proves the lemma. [Notice here that, the proof is still valid if Y happens to be A].

Theorem : there are no permanent 4-hop conflicts.

Proof : by contradiction.

Assume there is a permanent 4-hop conflict which can be represented by the topology: A{1}–B{2}–C{3}–D{4}–E{1}. Then according to *lemma 2*, there exist nodes X and Y such that: X{4}–Y–B{2}–C{3}–D{4}–E{1} and Y is MPR of B.

Now by noticing that a subgraph of this topology is: X{4}–Y–B{2}–C{3}–D{4}–...(a 4-hop conflict with address 4), we can apply *lemma 2* on this topology again: thus, there exist nodes U and V such that: U{3}–V–Y–B{2}–C{3}–D{4} (and V is MPR of Y) is part of the graph.

Now notice that the topology includes the subgraph of a 4-hop conflict with address 3: U{3}–V–Y–B{2}–C{3}, but this time with the noteworthy fact that Y is MPR of B{2} (this fact comes exclusively from the first application of *lemma 2*). But this fact is in contradiction with the *lemma 1* applied to the 4-hop conflict between U{3} and C{3}: indeed the *lemma 1* indicates(proves) that Y shall choose neither V, nor B{2} as MPR. The contradiction shows that the hypothesis that "there is a permanent 4-hop conflict" is impossible, hence the theorem.

Case of d $\geq$ 5

We have just shown that all 4-hop conflicts are resolved. Thus after a given time, there will not be any 2-hop conflict remaining for the MPR selection on the route between two nodes at five hops aways or more. Thus the classical rules of the MPR-flooding are sufficient to ensure that the MAD messages of two nodes at distance $d \geq 5$ will be exchanged between these two nodes. In other words, conflict between nodes at distance $d \geq 5$ are thus detected with MAD messages.

To be completly rigorous we have shown the previous results with the hidden assumption that when a conflict is detected, it is then successfully resolved. Thus to obtain the previous results we have to detect and resolve the conflicts at distance $d = 2$, then $d = 3$, then again $d = 4$ and finally at distance $d \geq 5$. This asssumption might not be true, if the resolution of a conflict leads to another conflict. This might happen when there is only a very small fraction of free available addresses in the pool. To overcome this problem we can use random address assignment. In such a case the process should eventually terminate with a network without address conflict.

4.2 Specification of the DAD-MPR Flooding algorithm

Let us recall the assumptions here.

Each node A periodically sends a MAD message M including:

- The originator address of A, $Orig_A$, in the OLSR message header.
- The message sequence number, $mssn$, in the OLSR message header.
- The node identifier ID_A (a string of bits) in the message itself.

The message is propagated by MPR-flooding to the other nodes ; but for DAD-MPR Flooding, the duplicate table of OLSR is modified, so that it also includes the node identifier list in the duplicate tuple(*Rule 1*). That is, a duplicate tuple, includes the following information:

- The originator address (as in OLSR standard duplicate table).
- The message sequence number (as in OLSR standard duplicate table).
- The list of node identifiers.

The detailed algorithm for DAD-MPR Flooding is the following:

- When a node B receives a MAD message M from node C with originator $Orig_A$, with message sequence number $mssn$, and with node identifier ID_A, it performs the following tasks:
  1. **If** a duplicate tuple exists with the same originator $Orig_A$, the same message sequence number, and ID_A is in the list of node identifiers, **Then**, the message is ignored (it has already been processed). The algorithm stops here.
  2. **Else** one of the following situations occurs :
     (a) A duplicate tuple exists with the same originator $Orig_A$ and the same message sequence number, but ID_A is not in the list of node identifiers: then, a conflict is detected (address $Orig_A$ is duplicated). ID_A is added to the list of node identifiers.

     (b) A duplicate tuple exists with the same originator $Orig_A$, but with a different message sequence number and ID_A is not in the list of node identifiers: then, a conflict is detected (address $Orig_A$ is duplicated). A duplicate tuple is created with the originator address, message sequence number and list of node identifiers containing only ID_A.

   (c) No duplicate tuple exists. A new one is created with the originator address, message sequence number and list of node identifiers containing only ID_A.

3. The MAD messages should be relayed by node B if one or more of the following rules are met:
   (a) Node C had chosen this current receiving node ,B, as a MPR.
   (b) The node C is the source of the MAD message i.e. it has the originator address $Orig_A$ (*Rule 3*).
   (c) One of the conflicting nodes is a 1-hop neighbor of the node detecting the duplication (*Rule 2*). In such a case, the TTL value of the MAD message showing the conflict is set to one before its retransmission. This also applies even if the current node has not been selected as a MPR by the previous message sender.

5 Initial address assignment and resolution of conflicts

5.1 Initial address assignment

There are two ways to allocate an address to an arriving node. The first way is to allocate a random address to this node and then to rely on the DAD algorithm to discover any potential address duplication. If this address is duplicated another random address will be chosen. This allocation process terminates when the selected address is conflict free. The second way is for the new node to ask for the help of one of its neighbors t o get an IP address. Since a configured node receives the MAD messages of all the nodes in the network, it can maintain a pool of not already used addresses. It can give such an address to the requesting node. In principle there will be no duplication with this scheme except if, due to MAD messages loss, the proposed address is duplicated or in the case of simultaneous requests in different locations of the network. These two schemes can be simply analyzed. Let us denote by N_n the number of nodes in the network. These nodes have a unique and non duplicated addresses. Let us denote by N_a the total number of addresses in the pool of addresses. In the first technique, the probablity that the chosen address will be duplicated is thus: $p = \frac{N_n}{N_a}$. If one denotes by D_1 the duration to detect a duplication, the average time to obtain a non duplicated address can be simply expressed as

$$\sum_{i\geq 1}(1-p)iD_1p^{i-1} = D_1\frac{1}{1-p} = D_1\frac{1}{1-p} = D_1\frac{1}{(1-\frac{N_n}{N_a})}$$

In the second scheme, to take into account the effect of transient packets loss we will suppose that a fraction h of the N_n configured nodes will not be

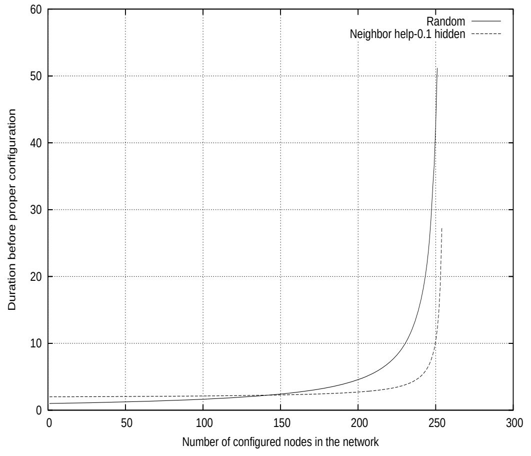


Figure 12: Duration for a first address assignment

known by a config ured node. Thus when a neighbor replies to a requesting node with an address, the pro bablity that the chosen address will be duplicated is: $p = \frac{hN_n}{N_a - N_n(1-h)}$. If one denotes by D_2 the duration to request an address and to detect a potential duplication, the average time to obtain a non duplicated address can be expressed as

$$\sum_{i \geq 1} (1-p) i D_2 p^{i-1} = D_2 \frac{1}{(1 - \frac{hN_n}{N_a - N_n(1-h)})}$$

In Figure 12 we show the duration for a requesting node to obtain a not duplicated address. To take into account, in the second scheme, the duration of the exchanges between the requesting node and one of its neighbors we have supposed that $D_2 = 2D_1$. We have also assumed that 10% of the MAD packets are lost, thus we can assume that $h = 0.1$. We are considering an address pool of 256 addresses; $N_a = 256$. We can see in Figure 12 that except when there are few configured nodes, the second approach offers better performances. When there are a few configured nodes the probabilty of choosing a duplicated address is small and the overhead induced by requesting an address to a neighbor is predominant. When there are numerous configured nodes, the probabilty of choosing a duplicated address increases and the second scheme performs better than the first.

5.2 Pool of addresses

The pool of addresses could be for local use only. For example, it could be reserved by the IANA authority for local MANET forwarding (i.e, those addresses must not be forwarded outside the MANET network, nor reached from outside). A second possibility consists in relying on some machines which will announce the prefix to use for address autoconfiguration for this MANET network. These machines could be connected to the internet, and act as gateways. In this case, the addresses may be global addresses, and could be seen from outside.

5.3 Resolution of a conflict

When two nodes A_1 and A_2 are configured with the same IP address and assuming that there is no packet loss, each of these two nodes will receive the MAD message of the other node. Thus the nodes where the conflict lies are bound to discover the conflict. A simple rule to solve this conflict will be: the node in conflict with the smallest identifier changes its address. Since this node knows via the reception of the MAD control messages the already assigned addresses, the new address must be selected at random among the addresses that are believed to be free. To be completely rigorous we can not be sure that all the addresses in the pool of non affected addresses are free. In fact, two nodes joining the network at almost the same time may be assigned the same IP address.

Moreover, if a fixed part of the first bits of the identifier is set to the "priority" of the node, this will lower the probability of "important" nodes having to change their addresses.

6 Control overhead of the DAD-MPR flooding algorithm

The overhead of the proposed autoconfiguration protocol can be rather easily evaluated. The main part of this overhead resides in the sending of MAD messages.

In this section, we show that the cost of MAD messages is bounded by the cost of other existing OLSR messages (HELLO messages and TC messages) multiplied by a given factor. Also we show that the factor is proportional to the MAD message rate. We then argue that the MAD message rate (fully adjustable) is expected to be one order of magnitude lower than the rate of other OLSR messages: this gives a hint about the reason why MAD message overhead is, at worst comparable, and in general much lower, than the cost of OLSR messages. In the last part, simulations are made.

6.1 Analytic bound of the cost of MAD messages

The overhead of the OLSR routing protocol without MAD messages is well known, and using results of [Viennot et al. 2002] (with a slightly different notation), the OLSR overhead as a number of bytes is given by:

$$overhead = T_h + T_t = \tau_h L_h N + \tau_t L_t o \rho N^2 \tag{1}$$

Variable	Meaning
δ	average degree of a node
N	number of nodes in the network
τ_h	Hello message rate
L_h	size of Hello messages
τ_t	TC message rate
L_t	size of TC messages
o	broadcast optimization factor, $\frac{1}{\delta} \leq o \leq 1$
ρ	proportion of nodes which are MPR of at least one node
τ_m	MAD message rate
L_m	size of MAD messages
T_h	total overhead of Hello messages (in bytes)
T_t	total overhead of TC messages (in bytes)
T_m	total overhead of MAD messages (in bytes)
L_a	size of one address
N_n	avg. number of neighbors of one node

Table 1: Notation used for control overhead

with the different parameters described in table 1.

The cost of MAD messages can be evaluated and bounded in a similar way: it is, at most, the cost of the retransmission of a MAD message by all the neighbors of one node due to Rule 3, plus the cost of retransmission of a MAD message by MPR-flooding to the entire network. More precisely we have:

$$T_m \leq \text{overhead of neighbors} + \text{overhead of MPR-flooding}$$
$$T_m \leq \tau_m L_m N N_n + \tau_m L_m N(oN)$$

It is possible to bound each term of this sum using the overhead of standard OLSR messages. Indeed, for the first term:

$$\tau_m L_m N N_n \leq \tau_m L_m N(\frac{L_h}{L_a}) = \frac{\tau_m}{\tau_h}\frac{L_m}{L_h}(\frac{L_h}{L_a})(\tau_h L_h N)$$

$$\tau_m L_m N N_n \leq T_h \frac{\tau_m}{\tau_h}\frac{L_m}{L_a}$$

and for the second term:

$$\tau_m L_m N(oN) = \frac{\tau_m}{\tau_t}\frac{L_m}{L_t}\frac{1}{\rho}(\tau_t L_t o \rho N^2)$$

$$\tau_m L_m N(oN) = T_t \frac{\tau_m}{\tau_t}\frac{L_m}{L_t}\frac{1}{\rho}$$

Hence the overhead of MAD messages is bounded by

$$T_m \leq \alpha T_h + \beta T_t \tag{2}$$

where

$$\alpha = \frac{\tau_m}{\tau_h} \frac{L_m}{L_a} \tag{3}$$

$$\beta = \frac{\tau_m}{\tau_t} \frac{L_m}{L_t} \frac{1}{\rho} \tag{4}$$

Or also, compared to the total standard OLSR overhead:

$$T_m \leq max(\alpha, \beta)(T_h + T_t) \tag{5}$$

For networks of similar density the optimization factor ρ is expected to be similar, hence α and β stay constant even when the area of the network grows.

Additionally, the rate of MAD messages τ_m can be adjusted: the overhead due to MAD messages is proportional to the MAD message rate. Hence there is a direct tradeoff between MAD message overhead and the promptness of duplicate address detection.

6.2 Discussion of the cost of MAD messages

In the following, the properties of α and β are discussed using intuition, but no proof: the section 6.3 will later justify and quantify the intuitions highlighted in this section.

The first observation is based on the fact that the DAD procedure may not need to be as fast to resolve conflicts as OLSR is fast to adjust to topology changes. Indeed, the standard OLSR messages should be sent with a rate at least of the order of magnitude of the rate of topology changes. The MAD messages should be sent with a rate at least of the order of magnitude of the rate of address changes or the rate of newcomers arrival in the network (or lower). Intuitively, in a MANET, there should be markedly more changes of topology per second than changes of addresses per second - or newcomers entry in the network per second. This translates into the fact that, arguably, the ratio of the rates $\frac{\tau_m}{\tau_h}$ and $\frac{\tau_m}{\tau_t}$ (appearing in the expressions of α and β) are both expected to be much lower than 1.

The second observation is, considering again the expression of α (equation 3) and β (equation 4), one can see that, for instance, α is the product of the previous (small) ratio of rates by the product of the ratio of sizes $\frac{L_m}{L_a}$. Now, considering that a MAD message includes an identifier which is a few times larger than an address, we estimate that the ratio would be larger than 1, typically 4 to 10. Arguably, the product of this second ratio (large but reasonnable) by the first one (quite small), could typically yield an α value still lower than 1 or close to 1.

Similarily for β, typical values of ρ are on the order 0.5 to 0.9, and hence, $\frac{1}{\rho}$ should be less than 2. Furthermore, the ratio of the size of the messages $\frac{L_m}{L_t}$ is even smaller that the previous ratio $\frac{L_m}{L_a}$, since a TC message includes several addresses. Hence this ratio, is typically of the order of magnitude of 1, and once again, the product of the ratio would yield a β which is much lower than 1.

Our conclusion is for a sensible choice of parameters, a rapid analysis of the structure of the equations 3 and 4 shows that one can expect both of them to be lower than 1 and hence have an MAD overhead lower than existing OLSR protocol overhead.

6.3 Simulation results

The overhead of the proposed autoconfiguration mechanism for $OLSR$, the overhead of MAD messages, has been simulated (using a simulator written in the Ocaml programming language) in order to evaluate its performance.

An idealized simulation model was choosen in order to evaluate precisely its impact: a given number of nodes are placed in a square area ; there is no mobility, and a unit disk graph[2] is used. There is no MAC, hence no contention or collision, and the transmission delay is uniform. The radio range is a parameter of the simulation.

6.3.1 Cost of MAD flooding

In our simulations, the first interesting parameter is the average cost of one diffusion of MAD message. Indeed, in MAD diffusion, not all, but only a fraction of the nodes will retransmit the MAD message, hence an interesting parameter is the fraction of the nodes of the network which retransmit on average a MAD message (in our graph, the fraction is expressed in percentage of the nodes of the network). Note that the size of one MAD message is fixed and that all nodes are diffusing MAD messages periodically, hence one can easily derive the absolute overhead counted in messages (or bytes) per second.

Figure 13 depicts **the proportion of the nodes which retransmit one MAD message**, for increasing density: the simulated network comprises 1000 nodes and for each simulation, a different radio range is set in order to modify the density of the neighboring nodes. The result of MPR-flooding is given on the same figure for reference and is actually the quantity oN of the section 6.1.

[2] In a unit disk graph, each node is identified with a disk of unit radius $r = 1$ in the plane, and is connected to all nodes within (or on the edge of) its corresponding disk.

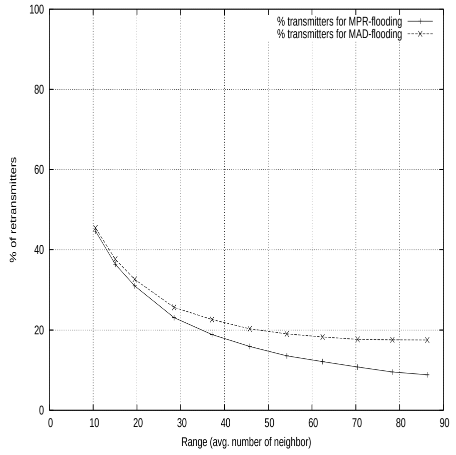


Figure 13: Fraction of the nodes in the network which retransmit the MAD message

As shown in that section, the evaluation of the MAD-flooding is a little more complex than just evaluating oN: indeed it is the overhead of the MPR-flooding, plus the extra transmission of the neighbors of the initial originator of the MAD message - which are not MPR. The average number of the neighbors which are not MPRs of a given node is thus expected to be: avg number of neighbors per node - avg number of MPRs per node. Hence an estimate would be:

$$\begin{aligned}\text{MAD-flooding overhead} &= \text{MPR-flooding overhead} \\ &+ (\text{avg number of neighbors per node - avg number of MPRs per node})\end{aligned}$$

In order to verify the estimate, using the previous MPR-flooding simulation results, we have calculated the corresponding MAD-flooding overhead. Precisely, for each simulation, we have two results: the overhead when actually simulating the MAD-flooding, and the overhead estimated from the MPR-flooding. Figure 14 depicts both the simulation results and the estimate. As they are close, the Figure 15 is more precise by depicting the ratio of the two quantities: we see that in our simulation set, the estimate is within 5 % of the simulation results, for the overhead of MAD message.

The general conclusion is that the additional overhead generated by MAD messages remains limited compared to one classical MPR-flooding: the cost of the MAD flooding is similar to the cost of MPR-flooding, with the exception that all the direct neighbors of the originator will retransmit (instead of only the MPR) in the first step.

It also appears that the estimate of the cost of the MAD-flooding computed

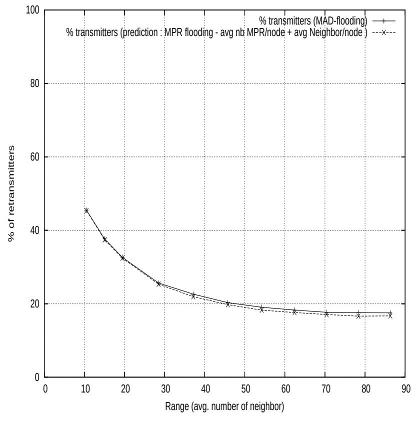


Figure 14: Comparison between actual results and prediction

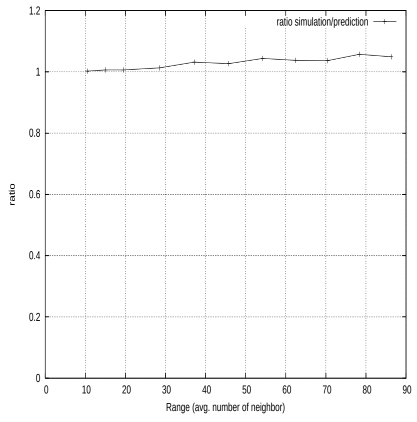


Figure 15: Ratio of overhead in simulation vs estimated overhead

from the measured cost of MPR-flooding is close the actual measured MAD-flooding, showing that the estimate is quite precise.

6.3.2 Comparison of the cost of MAD overhead with OLSR overhead

In the previous section, the cost of MAD-flooding was given, by simulation, as the percentage of nodes in the network that retransmit one MAD message.

This allows to compute the cost of MAD messages with an absolute value in bytes per second. But another interesting result is the relative cost of MAD messages compared to standard OLSR messages, in the spirit of the analytical results of section 6.1.

In order to do so, more parameters have to be defined. The following are used:

- Hello message rate, $\tau_h = 0.5\ s^{-1}$ (as specified in RFC 3626)
- TC message rate, $\tau_t = 0.2\ s^{-1}$ (as specified in RFC 3626)
- MAD message rate, $\tau_t = 1/60\ s^{-1}$ (one per minute)
- Size of address, $L_a = 4$ bytes
- Size of identifier, 16 bytes (hence $L_m = L_a + id_{size} = 20$)

Furthermore, the overhead of MAC, IP, UDP or OLSR message headers is also not taken into account.

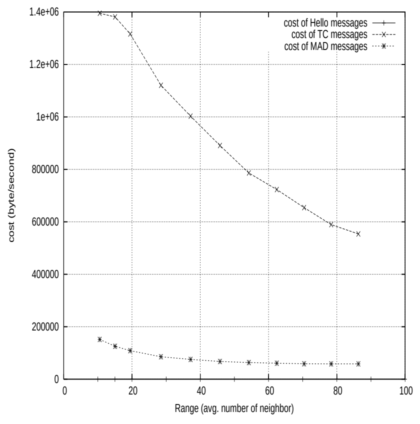


Figure 16: Cost of control messages in bytes

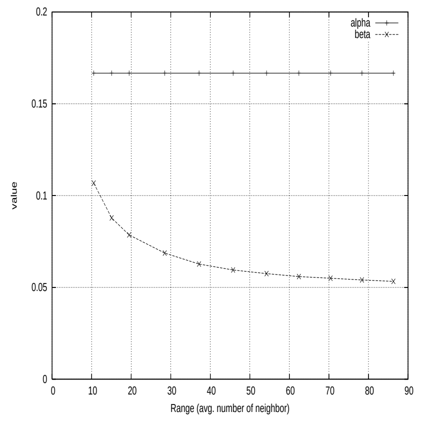


Figure 17: Evaluation of parameters α and β

The results, using the same simulations as in previous section 6.3.1, are represented on Figure 16 [3]. An observation is that in this network the TC message overhead dominates and Hello message overhead is insignificant. More importantly, that in comparison, the MAD message overhead is limited.

Furthermore, the bound in equation 2, is dominated by the part due to TC messages (with factor β). As a consequence, β is almost exactly the ratio between MAD message cost and TC message cost ; this is confirmed by calculation performed with simulation results. The Figure 17, shows the values of α and β for the corresponding simulations. The value of β appears to stay between 0.05 and 0.12, which indicates that, for these simulations, MAD message overhead is at most 5 % to 12 % of the TC message overhead (which is the main part of the OLSR protocol overhead), computed in bytes, something which is quite reasonnable and show the efficiency of the proposed DAD algorithm.

7 Conclusion

The autoconfiguration procedure proposed in this article relies on an efficient and proven duplicate address detection algorithm. A special control message MAD (Multiple Address Declaration) conveys a random identifier with the address of the node. This control message uses the OLSR genuine MPR-flooding algorithm to reach all the nodes in the network, however special rules have been added to ensure that even with address duplications the MAD messages will be propagated throughout the entire network. A formal proof of correctness of our duplicate

[3] The curve for the cost of HELLO messages is on the X axis.

address detection scheme is given in this paper. Simple approaches to allocate an address to a newly arriving node or to solve conflicts are also provided. Finally, a detailed analysis of the overhead induced by our autoconfiguration algorithm (MAD message overhead) is presented.

References

[Boudjit et al. 2004] Boudjit, S., Laouiti, A., Minet, P., Adjih, C.: "OLSR for IPv6 networks"; Proceedings of the 3^{rd} Med-Hoc-Net Workshop. Bodrum - Turkey (June 2004).

[Thomson and Narten 1998] Thomson, S., Narten, T.:"IPv6 Stateless Address Autoconfiguration", IETF RFC 2462, (Dec 1998).

[Droms et al. 2003] Droms, R., Ed., Bound, J., Volz, B., Lemon, T., Perkins, C., Carney, M.: "Dynamic Host Configuration Protocol for IPv6 (DHCPv6)", IETF RFC 3315, (July 2003).

[Jacquet et al. 2003] Jacquet, P., Muhlethaler, P., Minet, P., Qayyum, A., Laouiti, A., Clausen, T., Viennot, L., Adjih, C.: "Optimized Link State Routing Protocol", IETF RFC 3626, (October 2003).

[Perkins et al. 2003] Perkins, C., Belding-Royer, E., Das, S.: "Ad Hoc On-Demand Distance Vector(AODV) Routing", IETF RFC 3561, (July 2003).

[Qayyum et al. 2002] Qayyum, A., Laouiti, A., Viennot, L.: "Multipoint relaying technique for flooding broadcast messages in mobile wireless networks", HICSS: Hawai Int. Conference on System Sciences, Hawai - USA, (January 2002).

[Perkins et al. 2001] Perkins, C., Malinen, J., Wakikawa, R., Belding-Royer, E., Sun, Y.: "IP Address Autoconfiguration for Ad Hoc Networks", Internet Draft, IETF Working Group MANET, Work in progress, (November 2001).

[Weniger and Zitterbart 2002] Weniger, K., Zitterbart, M.: "IPv6 Autoconfiguration in Large Scale Mobile Ad-Hoc Networks", Proceedings of European Wireless 2002, Florence - Italy, (Feb 2002).

[Weniger 2003] Weniger, K.: "PACMAN : Passive Autoconfiguration for Mobile Ad Hoc Networks", Proceedings of IEEE WCNC 2003, New Orleans - USA, (March 2003).

[Nesargi and Prakash 2002] Nesargi, S., Prakash, R.: "MANETconf: Configuration of Hosts in a Mobile Ad Hoc Network", InfoCom 2002, (June 2002).

[Misra et al. 2001] Misra, A., Das, S., McAuley, A., K.Das, S.: "Autoconfiguration, Registration, and Mobility Management for prevasive Computing", IEEE Personal Communication, (August 2001), 24-31.

[Zhou et al. 2003] Zhou, H., M. Ni, L., W.Mutka, M.: "Prophet Address Allocation for Large Scale MANETs", IEEE INFOCOM 2003, (March 2003).

[Tanenbaum 1996] Tanenbaum, A, S.: "Computer Networks", Prentice Hall, 1996.

[Sayrafiezadeh 1994] Sayrafiezadeh, M., "The Birthday Problem Revisited", Math. Mag. 67, (1994), 220-223.

[Viennot et al. 2002] Laurent Viennot, Philippe Jacquet, Thomas Heide Clausen. : "Analyzing control Traffic Overhead in Mobile Ad-hoc Network Protocols versus Mobility and Data Traffic Activity", Proceedings of IFIP Med-Hoc-Net 2002, (June 2002).