# An Experimental Analysis of Routing Inconsistency in Indoor Wireless Mesh Networks

Rodrigo S. Couto, Miguel Elias M. Campista, Luís Henrique M. K. Costa, and Otto Carlos M. B. Duarte
Grupo de Teleinformática e Automação - PEE/COPPE - DEL/POLI
Universidade Federal do Rio de Janeiro - Rio de Janeiro, Brazil
P.O. Box 68504 - 21941-972, Rio de Janeiro, RJ, Brazil
E-mails:{souza,miguel,luish,otto}@gta.ufrj.br

***Abstract*— As of today, many routing protocols for wireless mesh networks have been proposed. Nevertheless, quite a few take the high loss rate of control packets into account. This work analyzes the problem of consistent routing information among wireless network nodes. To accomplish this, we propose a metric to evaluate the level of inconsistency among routing tables. Our experimental analysis demonstrates that the high loss rates seen in indoor environments negatively influence route computation. In addition, we demonstrate that the high network dynamics leads to severe instability in next hop selection. Results show that the effect of loss is significant and that the simple manipulation of routing protocol configuration parameters may be not enough to cope with the problem.**

## I. Introduction

Wireless mesh networks use a stationary backbone composed of wireless routers to improve the network connectivity and extend the coverage to users not within direct range of wireless gateways. In these networks, data forwarding is performed via multiple hops where backbone routers are the intermediate nodes.

In multihop communications routing plays a fundamental role. Routing protocols usually run shortest path algorithms to find the best path between any source-destination pair in the network [1]. In link-state routing protocols, the shortest path is computed from global topology information, maintained by each network node locally. This information is received from neighbor nodes and also from other nodes via periodical flooding of control messages. One of the main challenges in wireless mesh networking is to guarantee a synchronized view of the network topology at all nodes. This problem stems mainly from the delay and loss of control messages. Distributed shortest path algorithms may not correctly converge because they require all nodes to compute paths based on the same routing information base. Otherwise, the shortest path found from one source-destination pair will not be the same if computed by different nodes. The lack of synchronization in topology information leads to inconsistencies on nodes' routing tables [2]. The consequences are routing loops and non-optimal paths, which in wireless networks represent a severe problem given all the bandwidth constraints and the shared medium access.

The problem of routing synchronization in wireless networks has been addressed in related work. Yang and Wang [3] evaluate the importance of using routing metrics suited to the routing protocol used. They advocate that routing metric, protocol, and also testbed environment must be taken into account to avoid synchronization problems. Zaidi and Landfeldt [4] benefit from the broadcast nature of wireless transmissions to monitor nodes which send inconsistent routing information. In their work, every node overhears the medium to verify if their information is correctly retransmitted by other nodes. If not, they send warnings to the originating node. Upon receiving a warning, the node must update its routing information or will not be considered by other nodes. Huang *et al.* [5] evaluate the impact on network performance of routing update frequency. They conclude that increasing the frequency of updates does not necessarily improve network performance. Campista *et al.* [6] propose a routing protocol that reduces control message overhead in wireless mesh networks. They propose an algorithm to control flooding based on the position of the network gateways.

Wireless mesh networks use routing techniques to cope with link failures and network delays. Reducing loss caused by routing table inconsistencies can leverage such techniques. Javadi *et al.* [7] propose a multipath routing protocol that defines a primary path to send packets and alternative paths to carry copies of the same packets. Thus, this proposal creates redundant flows sent by multiple paths to improve network reliability. Nevertheless, the topology information can be quite inconsistent and the alternative paths can also lead to routing loops and loss. This solution is similar to the anypath routing, in which every two nodes are connected through multiple paths. The main difference, however, is that a node chooses only one of these paths to forward a single packet, depending on the amount of packet loss of each path. Laufer *et al.* [8] propose the multirate anypath routing protocol which also chooses the best transmission rate each node can use to forward a packet, further improving network performance.

In this work, we evaluate the problem of routing synchronization in wireless mesh networks. Unlike previous work, we propose a metric to evaluate the synchronization, i.e. the level of inconsistency among nodes' routing tables. By using this metric, it is possible to understand problems such as low delivery rate and routing loop existence in mesh networks. The analysis is based on a testbed operating with OLSR (Optimized

Link-State Routing) [9], which is a routing protocol often used in wireless mesh networks. We conduct experimental analysis in an indoor testbed located in our university campus. Results show that the effect of loss is significant and that the simple manipulation of routing protocol configuration parameters is not enough to avoid poor performance.

This work is organized as follows. Section II overviews relevant characteristics of OLSR. Section III introduces our proposed metric. Our experimental setup is described in Section IV and our results are presented in Section V. Finally, Section VI concludes this work and discusses future directions.

## II. OLSR Protocol

OLSR periodically sends control messages to maintain paths to all possible destinations in the network. This avoids route discovery procedures, but an amount of the network capacity is spent with control traffic. A variant of OLSR is under standardization in the upcoming standard IEEE 802.11s for wireless mesh networks. In addition to adjacency discovery, OLSR also uses `HELLO` messages to compute link states. It is also possible to infer two-hop neighbors from `HELLOs` because each node lists all its neighbors on these messages.

OLSR uses Topology Control (TC) messages to flood link states via broadcast. Thus, all neighbor nodes receive a TC message from a single transmission of the originating node. Nevertheless, since the same message can be received from multiple nodes, e.g. by the originating and a retransmitting node, OLSR uses a controlled-flooding mechanism to reduce redundant messages. Each node selects a subset of its neighbors, called the MPR (Multi-Point Relay) set, which is enough to reach all two-hop neighbors. Therefore, each node has its TC messages forwarded only by nodes within its own MPR set. The OLSR implementation `oslrd` used in this work has the configuration parameters `TcRedundancy`, `MprCoverage`, and `LinkQualityFishEye` to adjust the amount of control traffic injected in the network [9].

- `TcRedundancy`: adjusts the amount of information on each TC message. This parameter defines three possible levels. In level 0, TC messages only inform link states between the originating node and the nodes that have selected it to be in the MPR set (MPR selector set). This level provides the minimum information needed to all nodes compute routes. In Level 1, TC messages contain, in addition to the link states of Level 0, link states between the originating node and its neighbors in its MPR set. In Level 2, all link states from a node to its neighbors are announced.
- `MprCoverage`: defines the number of nodes in the MPR set that must be used to reach all two-hop neighbors. This parameter can assume any integer value from one to seven. If `MprCoverage` is equal to one, the control traffic is kept at the minimum. On the other hand, if `MprCoverage` is equal to $m$, each node selects its MPR set to guarantee that all its two-hop neighbors are reached by at least $m$ neighbors in the MPR set, if possible. Practically, the greater the `MprCoverage` value, the more control traffic is sent.
- `LinkQualityFishEye`: defines whether the flooding control mechanism Fisheye is used. This mechanism was proposed by Pei *et al.* [10] to reduce the number of topology control messages in ad hoc networks. Pei *et al.* advocate that the accuracy of TC messages is lost as the number of hops between any pair of nodes increases. Hence, it is more efficient to concentrate topology control messages among nodes closer to the originating node. To control TC message dissemination, Fisheye sets the TTL (*Time-To-Live*) field of IP.

The impact of these parameters is evaluated in Section V.

## III. Definitions and Notations

In this work, a wireless mesh network is modeled as a weighted connected graph $G = (V, E, e)$, where $V$ is the vertex set, $E$ is the edge set, and $e$ is the edge weight function. In this graph, a vertex denotes a backbone router, an edge denotes a wireless link connecting two routers, and the edge weight represents the link cost. The set of routers and links are fixed whereas the link costs vary over time. In addition, the set of edges is directed because the link cost may be different on each direction.

A path in $G$ is a sequence of different nodes where any consecutive pair is connected by a link. The cost of a path is the sum of all individual link costs within that path. Before forwarding a packet, an intermediate node $i \in V$ chooses its neighbor $v$ which provides the shortest-cost path toward a destination node $d \in V$. Hence, each node $i$ has a *forwarding table* mapping a destination node $d$ to a neighbor node $v \in V$. Let $N_i$ be the set of neighbors of node $i$, the forwarding table of $i$ ($f_i$) can be denoted as a function $f_i : d \rightarrow v \in N_i$.

Nodes must share exactly the same link state information to compute equivalent shortest paths and consequently to guarantee the same forwarding table. The set of link states known by a node $i$ is called the *topology map* ($M_i$) of this node. Let $S_i$ be the subset of $M_i$ composed of the link states from node $i$ to its neighbors ($N_i$), $R_i$ be the subset of $M_i$ composed of all the link states received by node $i$ from other network nodes, and $R_{i,j}$ be the subset of $R_i$ of the link states specifically from node $j$. Therefore, the topology map known by $i$ can be computed by $M_i = S_i \bigcup R_i$, where $R_i = \bigcup_{j=1, j \neq i}^{|V|} R_{i,j}$. Since the topology map is dynamic, the notation can be extended to $M_i^t = S_i^t \bigcup R_i^t$.

It is important to note that the subset $R_{i,j}^t$ can be seen as an estimation of the link states from node $j$ to its neighbors ($N_j$). As the link states received were produced in previous intervals of time, the routing table computed by $i$ may contain sub-optimal paths or lead to routing failures because they may not reflect the current status at $t$. This problem occurs as a consequence of the medium access method and the multihop communications which can delay the reception of link states from other network nodes, or even because control messages were lost. The goal of the Inconsistency Level metric is to quantify this problem.

**Definition 1 (Inconsistency Level)** We define as the reference topology map ($M_R^t$) the set of actual link states between the network nodes and its neighbors in a given instant $t$. Hence, $M_R^t = \bigcup_{i=1}^{|V|} S_i^t$. The level of inconsistency in the network is defined as the difference between the link states within the subset $R_{i,j}$ of the topology map of a given node $i$ ($M_i^t$), and the corresponding links in $M_R^t$. Therefore, the Inconsistency Level of the link states known by a node $i$ ($L_i$) is defined as:

$$L_i = \sum_{j=1, j\neq i}^{|V|} |R_{i,j}^t - S_j^t|. \quad (1)$$

If a given link state is not present in one of the topology maps, we consider that link cost infinity. In $L_i$ evaluation, however, we zeroed this link cost. Although it seems contradictory, if another value was used, the metric value $L_i$ would be predominantly the value of a single inconsistency. In our results, we will show the impact of this assumption. To compute $L_i$ for each node we need to have the topology map of all nodes considered in the evaluation. For this reason, $L_i$ metric is computed offline using all the topology maps logged in each node during a certain period of network operation.

## IV. Experimental Setup

Our testbed is deployed at the Federal University of Rio de Janeiro (UFRJ), Brazil. Our mesh network is composed of fourteen nodes: one PC and thirteen wireless routers. Nodes are placed inside rooms on the third floor of the building. Figure 1 illustrates the testbed. The PC is identified by $C$ while the routers are identified by $R_i$, where $i$ is the last octet of the router's IP address. In Room A, some nodes are placed in the first or in the second floor of a mezzanine. Numbers in superscript indicate if the router is on the first ($R_i^1$) or on the second floor ($R_i^2$) of the mezzanine. Nodes location is chosen to maintain the network connectivity and, at the same time, to maximize the number of available routes and hops.

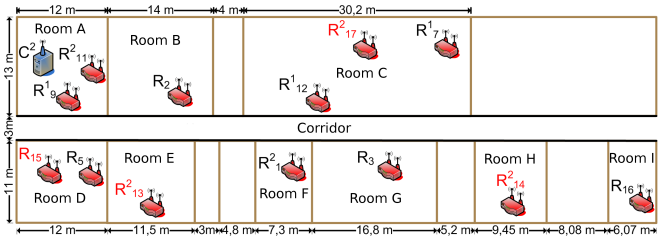



Fig. 1. Testbed topology.

The farthest nodes (nodes $C$ and $R_{16}$) are approximately 75 m away from each other. Room A is divided into smaller rooms by wood office partitions. Rooms from A to C and rooms from D to I are separated by simple masonry walls. The walls between all rooms and the corridor are double and made of masonry. Our testbed nodes operates using the IEEE 802.11g protocol at channel 6 (2437 MHz). There are other wireless networks in the area. In the worst case, we have found four networks, one of them using the same channel as our testbed.

### A. Basic Configuration

The PC ($C$) is equipped with Netgear IEEE 802.11 PCI card based on the Atheros AR5212 chipset. The thirteen IEEE 802.11 wireless routers are Linksys: six WRT54G, six WRT350N, and one WRT150N. The operational system used in our routers is Linux OpenWrt Kamikaze and the PC uses Linux Debian 3.1.

The advantage of using the WRT350N model is the possibility to store traces. This model has an USB interface used to enlarge routers' non-volatile memory. All equipment run `olsrd` version 0.5 [11]. The PC uses the Madwifi driver version 0.9 and the routers use the `Broadcom-drv` driver. None of the devices have directional antennas.

In this work, we use six different OLSR configurations combining `TcRedundancy`, `MprCoverage`, and `LinkQualityFishEye` parameters (Table I). The main goal is to analyze the impact of the different configurations in our testbed by comparing the obtained results. The level of routing control messages redundancy decreases with the configuration number. Some practical works [12] recommend using a high level of redundancy to improve the reception probability of control messages by network nodes, given the high loss rate of the wireless medium. The increase of control traffic, however, can result in a tradeoff because it reduces network resources that could be used to forward data traffic.

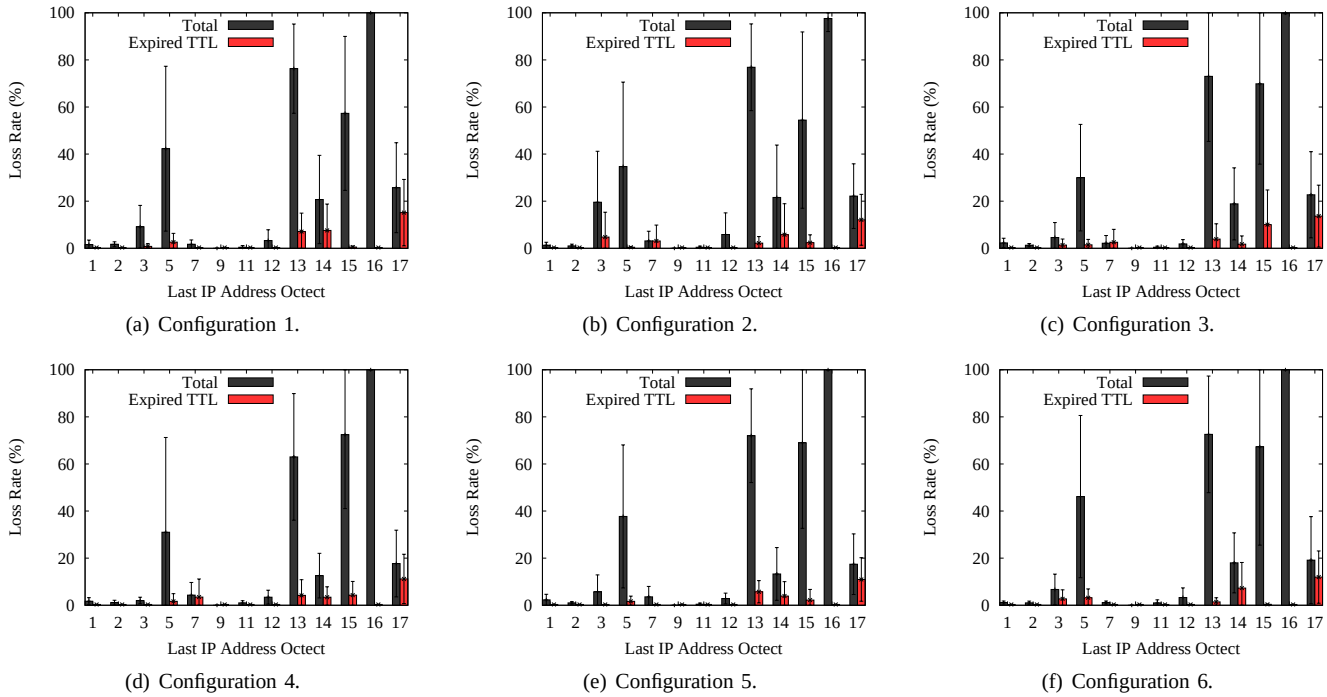
TABLE I
Different used configurations of OLSR.

| Configuration | `Tc Redundancy` | `Mpr Coverage` | `LinkQuality FishEye` |
|---|---|---|---|
| 1 | 2 | 7 | false |
| 2 | 2 | 7 | true |
| 3 | 1 | 1 | false |
| 4 | 1 | 1 | true |
| 5 | 0 | 1 | false |
| 6 | 0 | 1 | true |

The `olsrd` implementation uses the ETX (Expected Transmission Count) routing metric, which is based on the estimation of the number of transmissions needed to successfully transmit a frame on a link. The window size used to compute ETX is 100 `HELLOs`. All other parameters use default values.

## V. Measurements

In our experiments, the PC ($C$) sends sequences of `pings` to each router in the network. Each sequence is composed of 300 `pings` of 64 B each, sent in intervals of 1 s. Between two consecutive sequences there is an interval of five minutes without data traffic. Therefore, the total duration of each test is ten minutes. To each destination router, we tested all the six configurations in the ascending order shown in Table I. After finishing all six configurations, the destination router is changed and the `olsrd` configuration resets to the first one. It is worth mentioning that all nodes share the same OLSR configuration during the experiment. After finishing all thirteen routers the same procedure is repeated all over again from the first router. The complete procedure is run seven times.




(a) Configuration 1. (b) Configuration 2. (c) Configuration 3.

(d) Configuration 4. (e) Configuration 5. (f) Configuration 6.

Fig. 2. Total loss rate and loss rate because of TTL expiration.

Section V-A provides experimental analysis using the experiment described before. The analysis of loss rate and loss rate by TTL (Time-To-Live) expiration uses the output generated by the `ping` tool with the IP Record Route option enabled. The analysis of inconsistencies among the routing tables of the different routers is limited to the routers with USB interface. Both the PC and routers run the `olsrd_txtinfo` *plugin* of *olsrd*. This *plugin* generates debug information of `olsrd` such as neighbor links, topology map, and routing table. The neighbor links provide the ETX values found to each neighbor and the topology map provides all other link costs received. This information was used to compute the our metric.

In this work, the debug information is sampled at 1 s intervals and stored in the USB flash drives in order to compute the results offline. The Inconsistency Level metric was evaluated only for the nodes that store the topology. These nodes are the source $C$ and the routers $R_{13}$ , $R_{14}$ , $R_{15}$ and $R_{17}$. The metric proposed in this work is able to quantify the amount of inconsistency among the different routing tables during the wireless mesh network operation.

### A. Results

This section first presents sanity check tests and then results for the network loss rate and the level of routing inconsistency.

*1) Sanity check:* The first measurement aims at validating the storage procedure of OLSR plugin information using the USB interface. We evaluate the impact of the time to write in the flash drive when the router receives different traffic rates. The goal is to check if information may be lost between consecutive written operations. In this test, we use the `iperf` tool using the PC as the client and one of the routers with USB interface as the server. While the `iperf` traffic is received by the router, plugin information was recorded in busy wait in the flash drive. We perform two runs of tests using `iperf` at different rates and measure the time needed to write the information. Although all tests are performed using 64-byte `pings`, we used in our sanity check the `iperf` tool to perform worst-case analysis. Results show a maximum latency of 300 ms in writing procedures. Based on these results, we choose the period of 1 second to record plugin information because it is substantially higher than the write time.

*2) Loss rate and route length:* Figure 2 depicts the loss rate of `pings` obtained with each one of the analyzed configurations. The X axis indicates the last octet of the IP address of each router in the testbed. Note that the loss rate obtained to each destination is similar, considering the confidence interval, independent of the OLSR configuration. This shows that, in our testbed, parameters variation is not enough to change the loss rate. Even in Configuration 1, the configuration with the highest level of redundancy, the problem related to the wireless medium results in a high loss rate. Figure 2 also shows the loss rate of `pings` due to TTL expiration. On each hop traversed by the packet, the TTL is decremented. When TTL reaches zero, the packet is discarded and an ICMP (Internet Control Message Protocol) message is sent back to the originating node reporting the failure and the reason. In our case, discarding a packet because of TTL expiration is only possible if a routing loop occurs between source and destination. This indicates that nodes have an desynchronized view of the network topology and that they do not compute the same path. Hence, the loss due to TTL expiration reflects routing instability because of

inconsistencies among routing tables. Based on Figure 2, we conclude that the loss rate because of TTL expiration is also independent of the configuration.

Figure 3 illustrates the route length in number of hops between the PC, which generates `pings`, and the destination router. This length is computed when `pings` are successfully delivered and there is no routing loops. We observe again that the route length does not vary with the OLSR configuration. In this paper, we only show the result for Configuration 1 because of lack of space. Note that the route length is not directly related to loss rate. Although Node 5 ($R_5$ in Figure 1) has a route length similar to Node 1 ($R_1$), it obtains a loss rate greater than Node 1. Hence, control message loss is not directly related to the number of hops between source and destination, but it also accounts for other factors such as obstacles and interference.

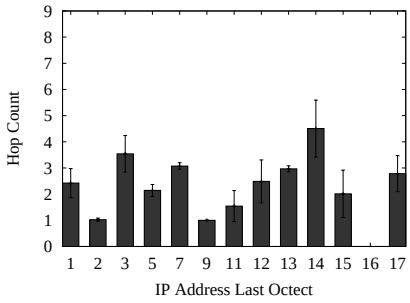



Fig. 3. Average path length between source and destination nodes of `ping` packets using Configuration 1.

*3) Inconsistencies of topology tables:* The level of inconsistency regarding the topology map of a given node is illustrated with a cumulative distribution function (CDF). The X axis represents the level of inconsistency observed in the interval of time considered. Figures 4 and 5 takes into account the results obtained in the seven runs of our complete experiment. In these figures, we plot the level of inconsistency. If it shows small values, this means that the inconsistencies among the topology information are not that high. Figures 4 and 5 plot the inconsistencies of the topology map of the source PC, Router 14 ($R_{14}$), and 17 ($R_{17}$). Figure 4 shows the results when the sequence of `pings` are triggered to Router 12 ($R_{12}$) whereas Figure 5 shows the results when they are triggered to Router 2 ($R_2$). The inconsistencies are computed to Router 14 because it is the farthest node from the source and to 17 because it is a router placed at an intermediate position. Although Routers 2 and 12 were chosen, the results obtained when other nodes receive `pings` do not show significant differences. We only use OLSR Configurations 1 and 6 because they represent, respectively, the one with the most and the least amount of control traffic. Note in Figure 4 that upon sending `pings` to Node 12, Configuration 1 presents higher inconsistency. Nevertheless, sending `pings` to Node 2, Figure 5, Configuration 1 presents lower inconsistency. Hence, it is observed that the simple manipulation of OLSR configuration parameters cannot guarantee a reduction of topology map inconsistencies.

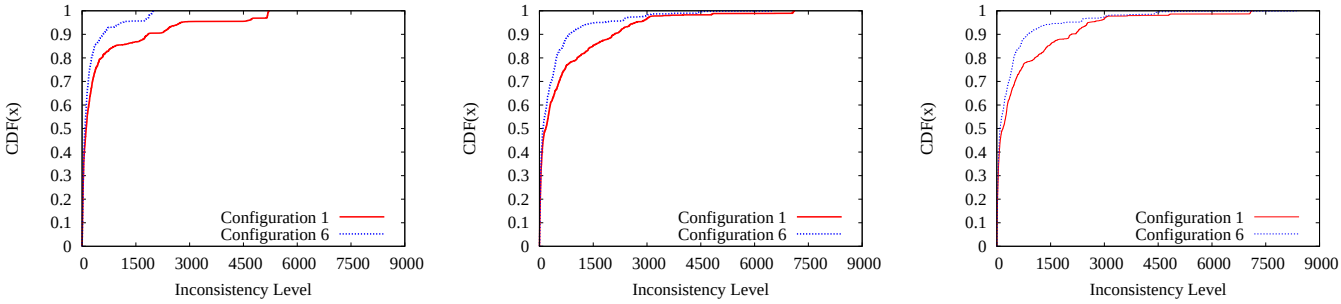
### *B. Number of inconsistent links*

The total number of inconsistencies between the reference topology and the analyzed topology is also evaluated. This analysis complements the results of Section V-A.3 which demonstrates that the high values of the proposed metric are a consequence of the number of inconsistencies among the compared topology maps. From these inconsistencies, we separated those without links with infinity ETX from those that present them. We use the infinity ETX when a link does not exist in the topology. The number of inconsistencies in the topology tables of a given node can be observed in Figure 6. This figure presents the probability density function (PDF) of the number of inconsistencies observed in Configurations 1 and 6, according to the view of the source and the Router 14. These plots are based on the results of all seven runs obtained in the `ping` test to Router 12. Note that there is a high number of inconsistencies with infinity ETX. This shows that links often break and emerge in the topology table. Therefore, considering the infinity ETX as zero avoids the $L_i$ metric to be predominantly dictated by the number of infinity ETXs.

Considering only the six nodes with USB, the maximum number of links which can be used for comparison is 78 since each node can have a link with 13 nodes in the network. Hence, it is observed that each time level of inconsistencies is computed, approximately 20% of the links have values different from the reference topology. This confirms that the inconsistency problem is severe and it is one of the main reasons why packet loss is frequent in wireless mesh networks.

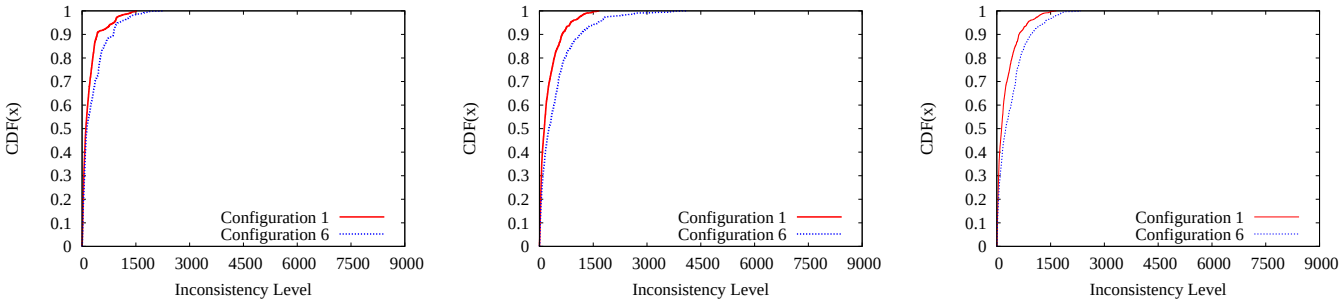
## VI. Conclusions

Many routing protocols have been proposed for wireless mesh networks. Nevertheless, few of them take into account the operation conditions of these protocols in real scenarios. In this work, the problem of routing inconsistency was analyzed. This problem is a direct consequence of control message loss which leads to nodes with different topology maps. To evaluate this problem, a testbed was used and the routing protocol OLSR (Optimized Link-State Routing) was employed to measure the impact of such inconsistencies under different protocol configurations. Results have shown that the inconsistency problem is not solved based on simple manipulation of configuration parameters. Depending on the conditions considered, the different configurations do not affect the network performance. Although the analysis was performed in an indoor testbed, in other scenarios, e.g. in an outdoor scenario, the inconsistency problem would be also relevant because loss of control messages can still happen. We verified that the inconsistency problem is severe and must be taken into account before proposing new protocols and new routing metrics. This problem can be minimized by using source routing or by adding positive acknowledgment to the routing protocol for topology control messages. As future work, we plan to extend our measurements and, later, to propose a new routing protocol that can combine techniques such as source routing, multipath routing, and positive acknowledgment.



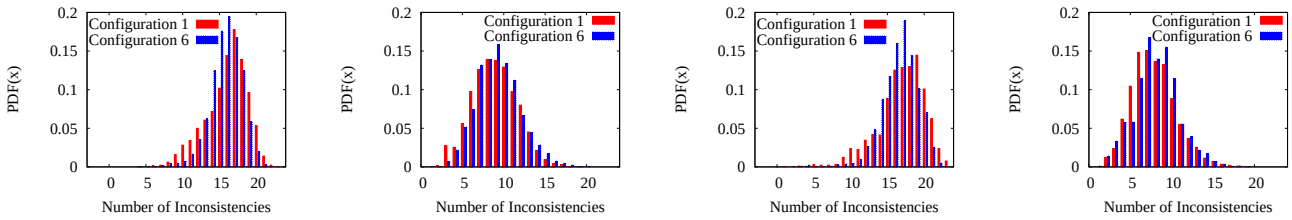

(a) Source. (b) Node 14. (c) Node 17.

Fig. 4. Topology map inconsistency upon sending `pings` to node 12.




(a) Source. (b) Node 14. (c) Node 17.

Fig. 5. Topology map inconsistency upon sending `pings` to node 2.




(a) Total - Source. (b) Infinity ETX - Source. (c) Total - Node 14. (d) Infinity ETX - Node 14.

Fig. 6. Number of inconsistent links.

## ACKNOWLEDGEMENT

This work was partially funded by CNPq, CAPES, COFECUB, FAPERJ.

## REFERENCES

[1] M. E. M. Campista, D. G. Passos, P. M. Esposito, I. M. Moraes, C. V. N. de Albuquerque, D. C. M. Saade, M. G. Rubinstein, L. H. M. K. Costa, and O. C. M. B. Duarte, "Routing metrics and protocols for wireless mesh networks," *IEEE Network*, vol. 22, no. 1, pp. 6–12, Jan. 2008.

[2] L. Wang, D. Massey, K. Patel, and L. Zhang, "FRTR: A scalable mechanism for global routing table consistency," in *International Conference on Dependable Systems and Networks (DSN)*, June 2004, pp. 465–474.

[3] Y. Yang and J. Wang, "Design guidelines for routing metrics in multihop wireless networks," in *IEEE Conference on Computer Communications (INFOCOM)*, Apr. 2008, pp. 1615–1623.

[4] Z. R. Zaidi and B. Landfeldt, "Monitoring assisted robust routing in wireless mesh networks," in *IEEE Internatonal Conference on Mobile Adhoc and Sensor Systems (MASS)*, Oct. 2007, pp. 1–6.

[5] Y. Huang, S. Bhatti, and S.-A. Sorensen, "Analysing the impact of topology update strategies on the performance of a proactive MANET routing protocol," in *International Conference on Distributed Computing Systems Workshops (ICDCSW)*, June 2007, pp. 13–20.

[6] M. Campista, L. Costa, and O. Duarte, "WPR: a proactive routing protocol tailored to wireless mesh networks," in *Global Telecommunications Conference, 2008. IEEE GLOBECOM 2008. IEEE*. IEEE, 2008, pp. 1–5.

[7] F. Javadi, K. Munasinghe, and A. Jamalipour, "Rapid and reliable routing mesh protocol (RRRMP)," in *Communications (ICC), 2010 IEEE International Conference on*. IEEE, 2010, pp. 1–5.

[8] R. Laufer, H. Dubois-Ferriere, and L. Kleinrock, "Multirate anypath routing in wireless mesh networks," in *INFOCOM 2009, IEEE*. IEEE, 2009, pp. 37–45.

[9] T. Clausen and P. Jacquet, "Optimized Link State Routing Protocol (OLSR)," IETF Network Working Group RFC 3626, Oct. 2003.

[10] G. Pei, M. Gerla, and T.-W. Chen, "Fisheye state routing in mobile ad hoc networks," in *IEEE Workshop on Wireless Networks and Mobile Computing (ICDCS)*, Apr. 2000, pp. D71–D78.

[11] "olsrd," http://www.olsr.org - Accessed in February/2010.

[12] https://list.open-mesh.net/pipermail/b.a.t.m.a.n/2008-June/000837.html - Accessed in February/2010.