

A simple pragmatic approach to mesh routing using BATMAN

David Johnson
Meraka Institute, CSIR
Pretoria, South Africa
Email: djohnson@csir.co.za

Ntsibane Ntlatlapa
Meraka Institute, CSIR
Pretoria, South Africa
Email: nntlatlapa@csir.co.za

Corinna Aichele
Berlin, Germany
Email: onelektra@gmx.net

Abstract—Development and performance analysis of ad hoc networking protocols has typically been performed by making use of software based simulation tools. However when running a routing protocol such as OLSR in large mesh network deployments, such as the 300 node Freifunk network in Berlin, it has been found that many of the optimization features, such as Multi-Point-Relays (MPRs), don't produce reliable routing. Some of the key issues which cause performance degradation with MPRs are routing loops due to asymmetrical links. In this paper a simple pragmatic routing protocol called BATMAN (Better Approach To Mobile ad hoc Networking) is presented as a response to the shortcomings of OLSR together with a comparison of its performance to OLSR. The experiments are run on a custom developed 7 by 7 grid of closely spaced WiFi nodes. The results show that BATMAN outperforms OLSR in terms of better throughput, less delay, lower CPU load and lower routing overhead.

I. INTRODUCTION

Mesh networking is a relatively new technology originating out of ad hoc networking research from the early 90's. As a consequence, there is still an ongoing effort to find routing protocols which perform best in large static or quasi-static wireless mesh networks.

Most of the protocols used for mesh networking grew directly out of protocols used for ad hoc networks which were designed with mobility in mind, examples of these protocols are Optimized Link State Routing (OLSR) [1], Dynamic Source Routing (DSR) [2] and Ad-hoc on-demand distance vector routing (AODV) [3] or may have been adaptations of these protocols to be more well suited to mesh networks such as Srcr [4] based on DSR and AODV-Spanning Tree (AODV-ST) [5] based on AODV.

The premise on which ad hoc networking protocols was built is very complex, one in which the network has a constantly changing topology due to mobility and losses over the wireless medium. A mesh network is a simpler subset of a general ad hoc network where little or no mobility is expected and only occasional route fluctuations should occur. However maximum throughput and minimum delay are far more important than just maintaining basic connectivity, which is often the best one can achieve when there is a high degree of mobility. With these foundational maxims, this paper presents a protocol called Better Approach to Mobile ad hoc Networking (BATMAN) which attempts to create a routing

protocol which learns routes using a very basic stigmergic approach.

Stigmergy is a term coined by a French biologist Pierre-Paul Grass in 1959 to refer to termite behavior. He defined it as the stimulation of workers by the performance they have achieved and is defined by the notion that an agents actions leave signs in the environment, signs that it and other agents sense and that determine their subsequent actions. For termites this is done by leaving pheromone trails that other termites sense to allow them to follow optimum routes to food or collectively build termite nests. A popular routing protocol which makes use of this phenomena is called AntHocNet [6] and BATMAN exhibits many similarities to the basic philosophy of this protocol.

In this paper a comparison is made of the performance of BATMAN and OLSR. The experiments are run on a custom developed 7 by 7 grid of closely spaced WiFi nodes. The use of testbeds for comparison of routing protocols is a recent phenomenon. A recent Network Test Beds workshop report [7] highlighted the importance of physical wireless test bed facilities for the research community in view of the limitations of available simulation methodologies. These mini scale wireless grids can emulate real world physical networks due to the inverse square law of radio propagation, by which the electric field strength will be attenuated by 6.02 dB for each doubling of the distance.

Traditionally ad hoc and mesh networking research has mostly been carried out using simulation tools but many recent studies [8] have revealed the inherent limitations these have in modelling the physical layer and aspects of the MAC layer. Researchers should acknowledge that the results from a simulation tool only give a rough estimate of performance. There is also a lack of consistency between the results of the same protocol being run on different simulation packages which makes it difficult to know which simulation package to believe.

Mathematical models are also useful in the interpretation of the effects of various network parameters on performance. For example, Gupta and Kumar [9] have created an equation which models the best and worst case data rate in a network with shared channel access, as the number of hops increases. However, recent work done by the same authors [10] using a real test bed, employing laptops equipped with IEEE 802.11

Standard (802.11) based radios, revealed that 802.11 multi hop throughput is still far from even the worst case theoretical data rate predictions.

In this paper we aim to:

- Describe the BATMAN protocol
- Briefly describe the working of the OLSR and highlight differences between OLSR and BATMAN
- Describe the mesh lab environment in which a comparison will be made between BATMAN and OLSR
- Analyse and compare the performance of the OLSR and BATMAN routing protocol on this testbed

II. BACKGROUND

This section will help provide some background to wireless mesh networking and the specific routing protocols that are discussed in this paper.

A. Ad hoc and mesh networks

An Ad hoc network is the cooperative engagement of a collection of wireless nodes without the required intervention of any centralized access point or existing infrastructure. Ad hoc networks have the key features of being self-forming, self-healing and do not rely on the centralized services of any particular node. There is often confusion about the difference between a wireless ad hoc network and a wireless mesh network (WMN).

A wireless ad hoc network is a network in which client devices such as laptops, PDA's or sensors perform a routing function to forward data from themselves or for other nodes to form an arbitrary network topology. When these devices are mobile they form a class of networks known as a mobile ad hoc network (MANET), where the wireless topology may change rapidly and unpredictably. Wireless sensor networks are a good example of a wireless ad hoc network.

A wireless mesh network is characterized by: dedicated static or quasi-static wireless routers which carry out the function of routing packets through the network, and client devices, which have no routing functionality, connecting to the wireless routers. Broadband community wireless networks or municipal wireless networks are good examples of wireless mesh networks.

All these types of ad hoc networks make use of ad hoc networking routing protocols which are being standardized by the IETF MANET working group [11]. There is also work being done to standardize mesh networking in the 802.11s standard [12].

B. BATMAN

BATMAN was born out of a response to the shortcomings of OLSR. A community wireless network based on OLSR known as Freifunk in Berlin noticed that OLSR had many performance shortcomings when the network grew very large (it is currently at about 300 nodes) [ref]. These included routes regularly going up and down due to route tables being unnecessarily flushed as a result of routing loops. There was a realisation that a routing algorithm for a large static mesh

needs to be developed from first principles and as a result the BATMAN project was started.

In BATMAN all nodes periodically broadcasts hello packets, also known as originator messages, to its neighbors. Each originator messages consists of an originator address, sending node address and a unique sequence number. Each neighbor changes the sending address to its own address and re-broadcast the message. On receiving its own message the originator does a bidirectional link check to verify that the detected link can be used in both direction. The sequence number is used to check the currency of the message. BATMAN does not maintain the full route to the destination, each node along the route only maintains the information about the next link through which you can find the best route.

C. System model

A network is modelled as $G = (N, E)$, where N represents a set of nodes and E represents a set of links between node pairs. For each node $i \in N$ in BATMAN, there exist a set of one-hop neighbours, K . The message from a source $s \in N$ to a destination d is transmitted along a link $(s, d) \in E$ if d is also an element of K otherwise it is transmitted along a multi-hop route made up of a link (s, i) and a route $[i, d]$, where i is a node in K and (s, i) is a link in E . The route $[i, d]$ represents a route from node i to node d through a subnet $S = (N - \{s\}, A - \{(s, i) : i \in K\})$.

D. Routing Objective

The objective is to maximize the probability of delivering a message. BATMAN does not attempt to check the quality of each the link, it just checks its existence. The links are compared in terms of the number of originator messages that have been received within the current sliding window.

E. Algorithm

- step 1 Consider routing message m from s to d on network G . Eliminate all links $(s, i) \forall i \notin K$ to reduce the graph.
- Step 2 Associate each link with weight w_{si} where w_{si} is the number of originator messages received from the destination through neighbour node i within the current sliding window.
- Step 3 Find the link with largest weight w_{si} in the sub-graph and send m along the link (s, i)
- Step 4 If $i \neq d$ repeat Steps 1 to 4 for routing message from i to d in the sub-graph S

Figures 1 through 3 illustrates the running of the above BATMAN algorithm for the following scenario:

- Node 1 want to send a message to Node 6. It only considers this set of links $\{(1, 2), (1, 3), (1, 4)\}$ to its neighbours $\{2, 3, 4\}$. The corresponding sets are illustrated in 2.
- Determine the best link as the link with higher the largest number of received originator messages from Node 6
- Suppose $(1, 2)$ is the best link then send message along this link.

- Since Node 2 is not the destination, reduce the graph N to graph S and repeat steps 1 to 4 of the algorithm. This is illustrated in 3
- Node 2 only considers this set of links $\{(2, 3), (2, 5)\}$ to its neighbours $\{3, 5\}$.
- Determine the best link as the link with higher the largest number of received originator messages from Node 6
- Suppose $(2, 5)$ is the best link then send message along this link.
- Since Node 2 is not the destination, reduce the graph N to graph S and repeat steps 1 to 4 of the algorithm.
- Node 5 only considers this set of links $\{(5, 6), (5, 3)\}$ to its neighbours $\{6, 3\}$.
- Determine the best link as the link with higher the largest number of received originator messages from Node 6
- Suppose $(5, 6)$ is the best link then send message along this link.
- Node 6 is the destination.

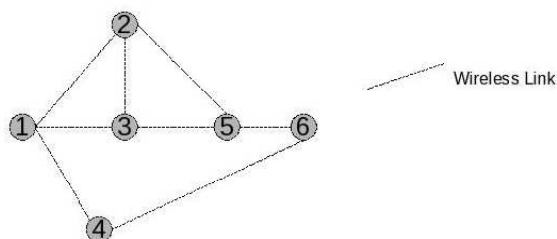


Fig. 1. Initial connected Graph G

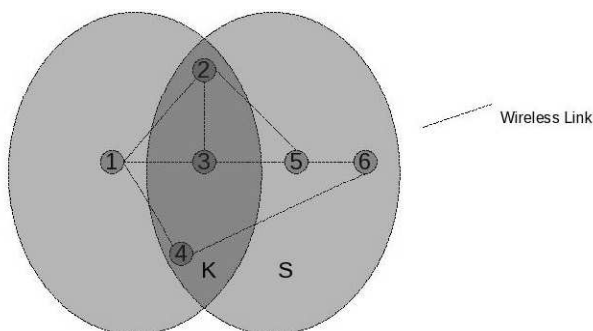


Fig. 2. Subsets of nodes formed by BATMAN algorithm in the 1st iteration. It shows the relationship between the three subset that are referred to in the above algorithm

The version of BATMAN used for all comparison in this paper is BATMAN 0.3-alpha.

F. Optimized Link State Routing (OLSR) protocol

Pro-active or table-driven routing protocols maintain fresh lists of destinations and their routes by periodically distributing routing tables in the network. The advantage of these protocols is that a route to a particular destination is immediately available. The disadvantage is that unnecessary routing traffic is generated for routes that may never be used. The Optimized

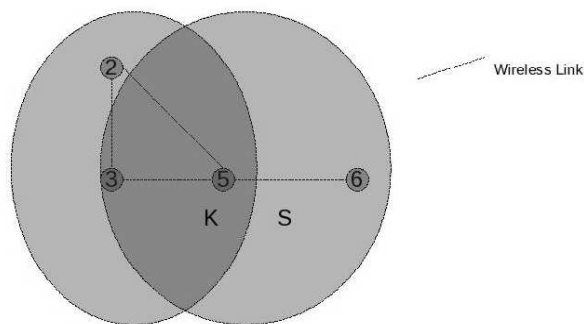


Fig. 3. Subsets of nodes formed by BATMAN algorithm in the 2nd iteration. It shows the relationship between the three subset that are referred to in the above algorithm

Link State Routing (OLSR) [1] pro-active routing protocol will be evaluated on the testbed in this paper.

OLSR reduces the overhead of flooding link state information by requiring fewer nodes to forward the information. A broadcast from node X is only forwarded by its multi point relays. Multi point relays of node X are its neighbors such that each two-hop neighbor of X is a one-hop neighbor of at least one multi point relay of X. Each node transmits its neighbor list in periodic beacons, so that all nodes can know their 2-hop neighbors, in order to choose the multi point relays (MPR)

Figure 4 illustrates how the OLSR routing protocol will disseminate routing messages from node 3 through the network via selected MPRs.

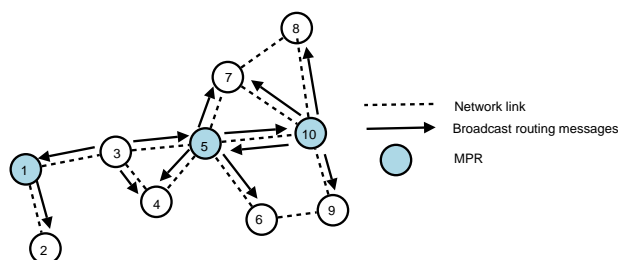


Fig. 4. OLSR routing protocol showing selection of MPRs

The OLSR source code that is run on the wireless grid can make use of two different types of routing metrics. The Request for Comments (RFC) for OLSR makes use of the hysteresis routing metric to calculate link quality between nodes. A new routing metric, called Expected Transmission Count (ETX) [13] proposed by MIT, has also been incorporated into the source code for OLSR but it is not officially part of the RFC. This is widely accepted to be a superior routing metric to basic hysteresis and is therefore used for all comparative analysis between OLSR and BATMAN.

ETX calculates the expected number of retransmission that are required for a packet to travel to and from a destination. The link quality, LQ , is the fraction of successful packets that were received by us from a neighbor within a window period.

The neighbor link quality, NLQ , is the fraction of successful packets that were received by a neighbor node from us within a window period. Based on this, the ETX is calculated as follows:

$$ETX = \frac{1}{LQ \times NLQ} \quad (1)$$

In a multi-hop link the ETX values of each hop are added together to calculate the ETX for the complete link including all the hops. Figure 5 shows the ETX values for 7 consecutive successful packets followed by 7 consecutive unsuccessful packets assuming a perfectly symmetrical link and a link quality window size of 7.

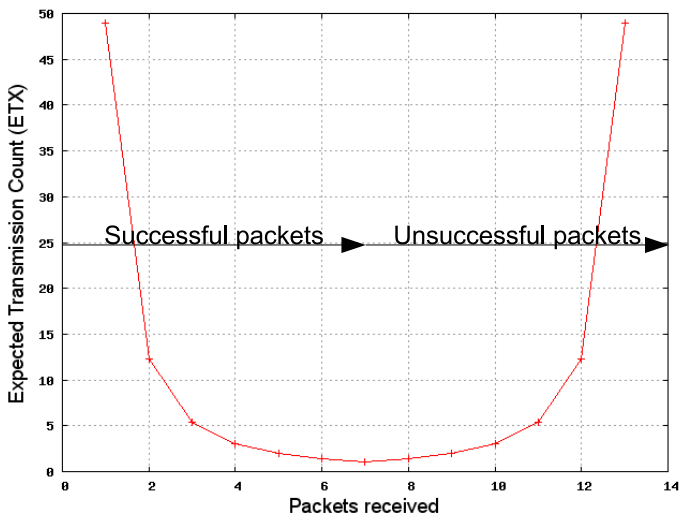


Fig. 5. ETX Path metric values for successive successful and unsuccessful packets

A perfect link is achieved when ETX is equal to 1. ETX has the added advantage of being able to account for asymmetry in a link as it calculates the quality of the link in both directions. Unlike Hysteresis ETX improves and degrades at the same rate when successful and unsuccessful packets are received respectively. Routes are always chosen such that the sum of all the ETX values of adjacent node pairs is minimized.

The Linux implementation of OLSR developed by Tonnesen [14] was used for comparisons. This implementation is commonly called olsr.org and is now part of the largest open source ad hoc networking development initiative. Version 0.5.5, which is RFC3626 compliant, is used and is capable of using the new ETX metric for calculating optimal routes as well as using an optimised version of the Dijkstra algorithm.

III. DESCRIPTION OF THE MESH TESTBED

The mesh testbed consists of a wireless 7x7 grid of 49 nodes, which was built in a 6x12 m room as shown in Figure 6. A grid was chosen as the logical topology of the wireless testbed due to its ability to create a fully connected dense mesh network and the possibility of creating a large variety of other



Fig. 6. Layout of the 7x7 grid of Wi-Fi enabled computers

topologies by selectively switching on particular nodes and to make repeatability of the experiment possible.

Each node in the mesh consists of a VIA 800 C3 800MHz motherboard with 128MB of RAM and a Wistron CM9 mini PCI Atheros 5213 based Wi-Fi card with 802.11 a/b/g capability. For future mobility measurements, a Lego Mindstorms robot with a battery powered Soekris motherboard containing an 802.11a (5.8 GHz) WNIC and an 802.11 b/g (2.4 GHz) WNIC shown in Figure 6 can be used.

Every node was connected to a 100 Mbit back haul Ethernet network through a switch to a central server, as shown in Figure 7. This allows nodes to use a combination of a Pre-boot Execution Environment (PXE), built into most BIOS firmware, to boot the kernel and a Network File System (NFS) to load the file system.

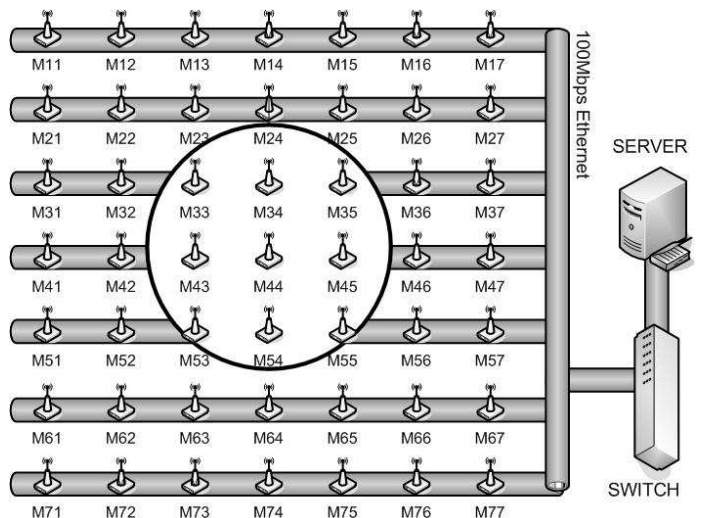


Fig. 7. The architecture of the mesh lab. Ethernet is used as a back channel to connect all the nodes to a central server through a switch. Each node is also equipped with an 802.11 network interface card.

The physical constraints of the room, with the shortest length being 7m, means that the grid spacing needs to be about 800 mm to comfortably fit all the PCs within the room dimensions.

At each node, an antenna with 5 dBi gain is connected to the wireless network adapter via a 30 dB attenuator. This introduces a path loss of 60 dB between the sending node and the receiving node. Reducing the radio signal to force a multi hop environment, is the core to the success of this wireless grid and this is discussed later.

The wireless NICs that are used in this grid have a wide range of options that can be configured:

- *Power level range:* The output power level can be set from 0 dBm up to 19 dBm.
- *Protocol modes:* 802.11g and 802.11b modes are available in the 2.4 GHz range and 802.11a modes are available in the 5 GHz range
- *Sending rates:* 802.11b allows the sending rate to be set between 1 Mbps and 11 Mbps and 802.11g allows between 6 Mbps and 54 Mbps

This network was operated at 2.4 GHz due to the availability of antennas and attenuators at that frequency, but in future the laboratory will be migrated to the 5 GHz range, which has many more available channels with a far lower probability of being affected by interference.

A more detailed analysis of the lab environment is available here [15]

IV. MEASUREMENT PROCESS

All measurements other than throughput tests were carried out using standard Unix tools available to users as part of the operating system. The measurement values were sent back to the server via the Ethernet ports of the nodes and therefore had no influence on the experiments that were being run on the wireless interface.

It was found that the lab provides the best multi hop characteristics trade off with the best delay and throughput when the radios are configured with the following settings:

- Channel = 6
- Mode = 802.11b
- Data rate = 11 Mbps
- TX power = 0 dBm

In order to avoid communication gray zones [16], which are illustrated in Figure 8, the broadcast rate is locked to the data rate. Communication grey zones occur because a node can hear broadcast packets, as these are sent at very low data rates, but no data communication can occur back to the source node, as this occurs at a higher data rate.

The following measurement processes were used for each of the metrics being measured in the ad hoc routing protocols:

- 1) *Testing under network load:* For the throughput, route flapping, delay, packet-loss and CPU and memory load tests, a set of 4 data paths was setup to continuously send 1500 bytes ping packets across the network in order to load the network. These were setup between the two

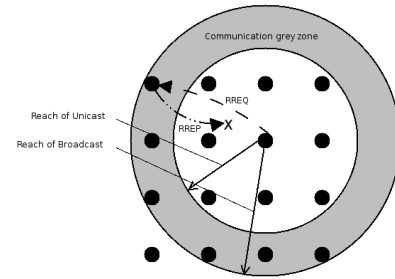


Fig. 8. Communication grey zones.

opposite corner nodes as well as between the nodes in the middle of the edges. Figure 9 shows the pairs of data paths that were set up.

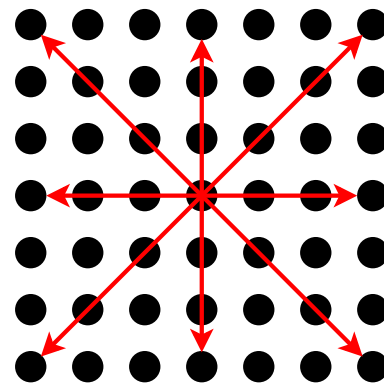


Fig. 9. Loading the network with ping traffic across the network.

- 2) *Delay:* Standard 84 byte ping packets were sent for a period of 10 seconds. The ping reports the round trip time as well as the standard deviation.
- 3) *Packet loss:* The ping tool also reports the amount of packet loss that occurred over the duration of the ping test
- 4) *Static Number of hops for a route to a destination:* The routing table reports the number of hops as a routing metric.
- 5) *Round trip route taken by a specific packet:* The ping tool has an option to record the round trip route taken by an ICMP packet but unfortunately the IP header is only large enough for nine routes. This sufficed for most of the tests that were done but occasionally there were some routes, which exceeded 9 round trip hops, and no knowledge of the full routing path could be extracted in these instances. However this was large enough to always record the forward route taken by a packet.
- 6) *Route flapping:* Using the ping tool with the option highlighted above to record the complete route taken by a packet every second, it is a simple process to detect how many route changes occurred during a set period of time by looking for changes in the route reports.

- 7) *Throughput*: The tool Iperf [16] was used for throughput measurements. It uses a client server model to determine the maximum bandwidth available in a link using a TCP throughput test but can also support UDP tests with packet loss and jitter. For these experiments an 8K read write buffer size was used and throughput tests were performed using TCP for 10 seconds. UDP could be considered a better choice as it measures the raw throughput of the link without the extra complexity of contention windows in TCP. This does make the measurement more complex, however, as no prior knowledge exists for the link and the decision on the test transmission speed is done through trial and error.
- 8) *Routing traffic overhead*: In order to observe routing traffic overhead the standard Unix packet sniffing tool tcpdump was used. A filter was used on the specific port that was being used by the routing protocol. The measurement time could be varied by the measurement script, but 20 seconds was the default that was mostly used. The tool made it possible to see the number of routing packets leaving and entering the nodes as well as the size of these routing packets. To force dynamic routing protocols such as AODV and DYMO to generate traffic while establishing a route, a ping was always carried out between the furthest two points in the network.
- 9) *Growing network size*: When tests are done which compare a specific feature to the growing number of nodes in the network, a growing spiral topology, shown in Figure 10, starting from the center of the grid, is used. This helps to create a balanced growth pattern in terms of distances to the edge walls and grid edges, which may have an electromagnetic effect on the nodes.

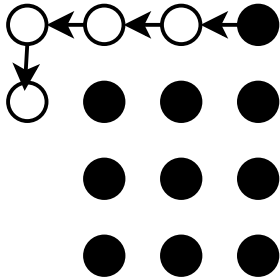


Fig. 10. Growing spiral topology for tests which compares a metric against a growing network size.

- 10) *Testing all node pairs in the network*: When throughput and delay tests were carried out on a fixed size topology, all possible combinations of nodes were tested. If the full 7x7 grid was used this equates to 2352 (49×48) combinations.
- 11) *RTS/CTS tuned off*: All tests are done with RTS/CTS disabled as this did not improve the performance of the mesh, other researchers have reported similar findings [17]
- 12) *CPU load and memory footprint*: In order to examine

the resources consumed by a routing protocol, the CPU load and memory footprint were analysed. The Unix *top* command was used. The cpu and memory consumption was analysed for 10 seconds and 1 second intervals and an average was reported.

V. RESULTS

Performance analysis of BATMAN and OLSR is now presented. The settings for each protocol was made as similar as possible in order for them to be fairly as possible, although each protocol has some features which the other does not have.

OLSR was used with the following settings

- HELLO interval = 1 second
- Topology Control (TC) interval = 1 second
- HELLO validity interval = 200 seconds
- TC validity interval = 100 seconds
- Fisheye = ON (TC messages are sent with 3 hop limit)
- Dijkstra limit: Ignore topology info from nodes ≥ 3 hops, update topology info every 3 seconds
- Linkquality (LQ) is used for MPR selection and routing
- LQ window = 100
- TC redundancy = send to all neighbours
- MPR coverage = 5 (i.e. up to 5 are selected to reach every 2 hop neighbour) this setting essentially disables the MPR optimization feature due to the problem with routing loops

BATMAN was used with the following settings

- HELLO interval = 1 second
- TTL = 50
- Windows size = 100

A. Routing overhead

The ability of a routing protocol to scale to large networks is highly dependent on its ability to control routing traffic overhead. Routing traffic contains messages that a routing protocol needs to establish new routes through a network, maintain routes or repair broken routes. For BATMAN these are only Originator messages (OGM's) and for OLSR these are HELLO messages as well as Topology Control (TC) messages. These are sent periodically to allow neighbouring nodes to learn about the presence of fellow nodes or they can be topology messages containing routing tables.

The amount of inbound and outbound routing traffic as well as the packet size of routing packets was measured as the network size grows in a spiral fashion. The measurement process was described in Section IV. Once this data was collected for each node in the network, the traffic was averaged across all the nodes in the network and normalized to the amount of traffic per second.

Figure 11 shows the inbound traffic for OLSR and BATMAN and Figure 12 shows the outbound traffic. Outbound traffic should always be less than the amount of inbound traffic as a node makes a decision to rebroadcast a packet or not. The rules for deciding whether to forward a routing packet are the following for BATMAN

- OGM's from single hop neighbours are always rebroadcast
- Only OGM's received by the best ranking neighbour are rebroadcast
- If the TTL has reached 0, the OGM is not rebroadcast
- OGM's are only rebroadcast to bi-directional neighbours, there is one exception to this rule when the network has no knowledge of its neighbours and it needs to test for bi-directionality.
- When a node receives an OGM, it first checks whether it already has received an OGM with the same originator and sequence number. If it has, then the OGM is discarded, without rebroadcasting

Outbound routing packets are the most important overhead to analyse as the routing protocol has control over this. OLSR mostly uses less routing packets as it has stricter rules for forwarding HELLO and TC messages. However above 45 nodes the number of routing packets overtakes BATMAN. BATMAN rises quickly to an average of 7.5 packets per node per second but then stays very constant.

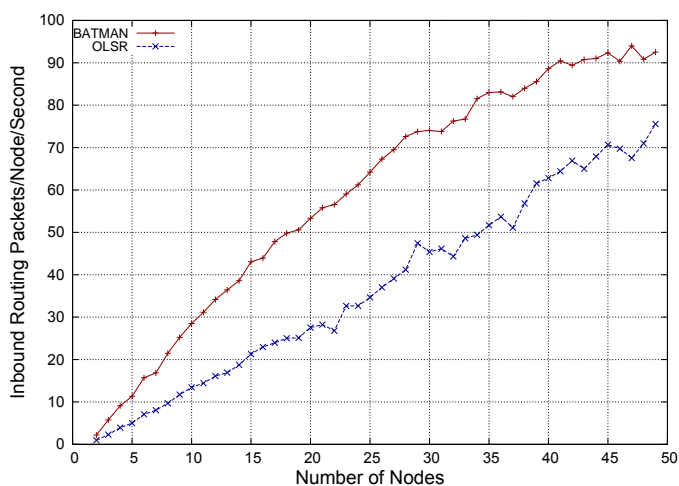


Fig. 11. Inbound routing packets per node per second versus increasing number of nodes using a growing spiral.

In order to know the true routing overhead of a routing protocol, the packet length needs to be known. Figure 13 shows how routing packet lengths grow as the number of nodes increase. This is another important characteristic to analyze if a routing protocol is to scale to large networks.

As the network grows, OLSR needs to send the entire route topology in Topology Control (TC) update messages, which helps explain this steady linear increase with the number of nodes. BATMAN on the other hand does not embed any routing information in the routing packets and therefore does not grow rapidly at all. In order to calculate the total overhead in terms of bytes per second for a routing protocol, the packet length is multiplied by the number of packets per second leaving a node. Carrying out this calculation for OLSR and BATMAN for the full 49 node grid reveals an overhead of 675 bytes per second for BATMAN and 6375 bytes per second

for OLSR, which reveals that OLSR has a routing overhead which is 10 fold that of BATMAN for network of this size.

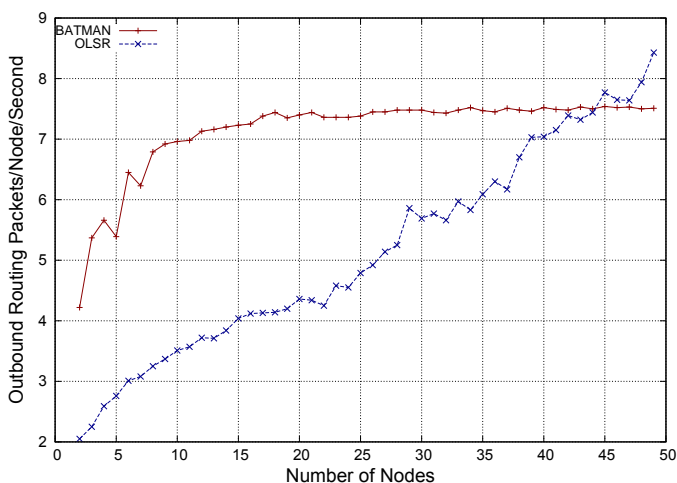


Fig. 12. Outbound routing packets per node per second versus increasing number of nodes using a growing spiral.

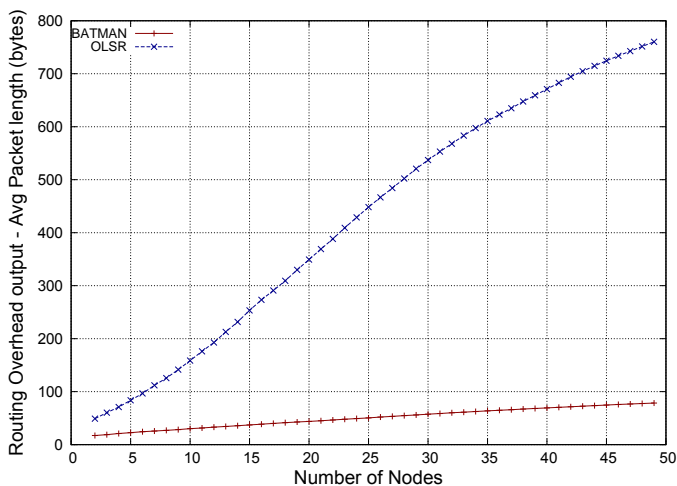


Fig. 13. Average Routing Packet length growth versus increasing number of nodes.

B. Throughput, packet loss, route flapping and delay measurements

The ability of a routing algorithm to find an optimal route in the grid will be exposed by its throughput, packet loss and delay measurements. Route flapping, which is an established phenomenon in wireless mesh networks [18], can also have a serious detrimental effect on the performance of the network.

The maximum network complexity was used to test which routing metric in OLSR performed the best under difficult conditions with thousands of alternative routes. Tests were carried out for all 2352 (49×48) possible pairs in the 7x7 grid and Table I highlights the averages for all the results.

These tests were also performed while the network was under load by starting 4 simultaneous data streams between across the network. (see Section IV)

TABLE I

Comparison of throughput, delay and packet loss for full 7x7 grid

Routing Protocol	Forward hop count	Symm links (%)	Seconds per Route change	Packet loss (%)	Delay (ms)	Throughput (kbps)	No link (%)
BATMAN	1.88	28	25.64	2.63	7.61	1378.35	1.11
OLSR	2.26	61	12.20	1.68	17.39	1177.92	0.60

BATMAN achieved the best overall throughput as well as the least delay with the least number of hops. The average amount of time to a route change was half that of OLSR which could account for its better throughput due to this route stability. It also had the smallest number of asymmetrical links which is symptomatic of a protocol which calculates routes based only on listening for originator messages from distant sources.

OLSR had about 1% less packet loss and about 0.5 % less broken links. This however did not translate into any advantage in terms of better delay or throughput and is not statistically significant. Double the amount of route flapping is one of the contributors to its weaker poor performance.

The following graphs take a closer look at how these protocols perform as the distance between the nodes increase.

A very clear relationship between route changes and distance is seen in Figure 14, which increases fairly linearly with OLSR beginning to level off after about 4 m. BATMAN clearly shows better route stability even at long distances in the grid.

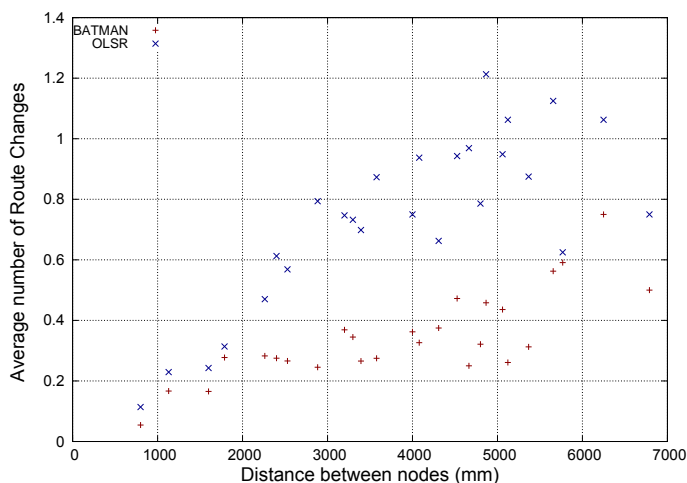


Fig. 14. Route changes versus distance for the OLSR and BATMAN protocol in the 7x7 wireless grid

Figure 15 shows the how hop count for BATMAN and OLSR which quickly diverges as the distance increases. OLSR's higher hop count creates more alternative routes to choose from, which will result in a higher degree of route flapping and a higher CPU load as will be seen later.

Figure 16 shows that BATMAN always has approximately 15% better throughput than OLSR over the full range of the grid. This shows that optimal routes are being found by BATMAN rather than OLSR for all distances. The decreases in

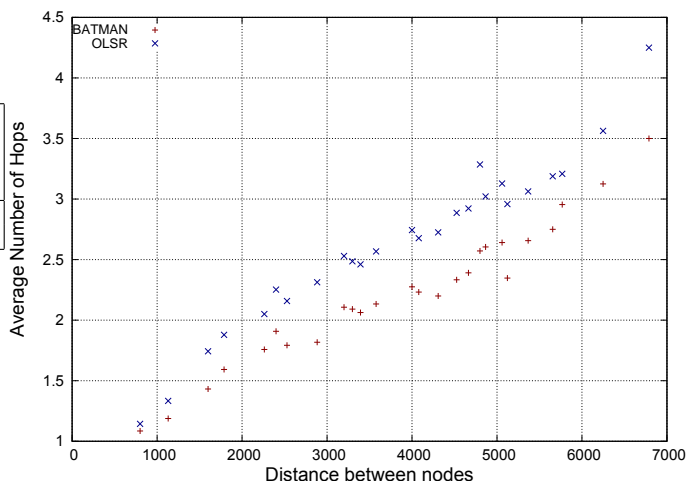


Fig. 15. hop count versus distance for the OLSR protocol in the 7x7 wireless grid

throughput follows a standard theoretical logarithmic drop-off ($\lambda(n) = \frac{W}{\sqrt{n \log(n)}}$) that is described by Gupta and Kumar's theoretical analysis of throughput degradation over multiple hops for ad hoc networks [9].

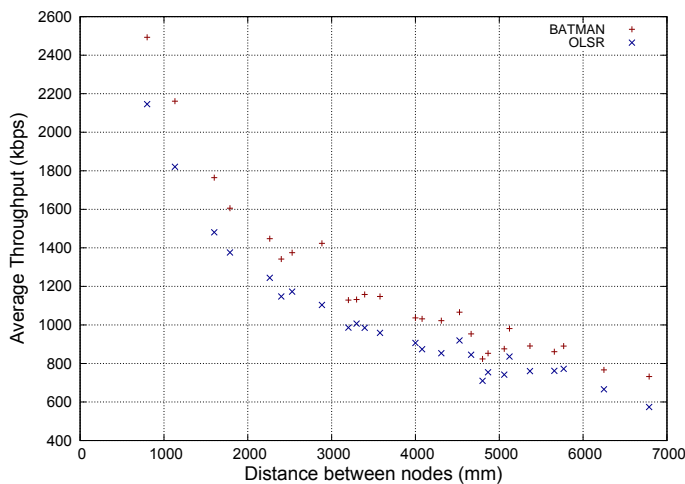


Fig. 16. Throughput versus distance for the OLSR protocol in the 7x7 wireless grid

These results have been carried out with OLSR fine tuned to perform optimally for a static mesh network where MPR optimization is disabled and timeouts are set to very long intervals. If these optimizations are not employed OLSR would have shown worse performance [19] due to some of the inherent weaknesses in OLSR such as routing loops as well as a high degree of route flapping. The ETX metric has also been shown to have inherent flaws when calculating the optimal route path by summing up ETX values of link pairs [19].

Some mechanisms are being developed to decrease BATMAN's routing overhead and therefore CPU load by aggregating routing messages, which would decrease BATMAN's overhead even further but this could begin to penalize its gains

on optimal throughput as well as minimal packet loss.

C. CPU load and memory consumption

The results for the amount of resources consumed as the network size is increased in a spiral fashion is now presented.

CPU load is directly affected by the number of packets it needs to process as well as the complexity of the algorithm needed to compute the optimal routes in the routing tables. The need to use integer or floating point mathematics in these algorithms also has a great impact. Although BATMAN exhibited a greater number of outbound routing packets than OLSR below a network size of 45, it proved itself to be far less CPU intensive than OLSR for network sizes greater than 6 as seen in Figure 17. At the full network size of 49 nodes OLSR was using 44% of a linksys's CPU compared to BATMAN which was only using 4%. The impact of OLSR's high CPU load is very serious, as it could saturate the ability of the router to handle routing packets or route data packets at fairly low network sizes of just over 100 nodes if the tendency is extrapolated. Some work is being done to remove floating point operations from OLSR's route calculations which would mitigate some of the CPU load for OLSR. Further comparisons will be carried out once this code is available.

It is interesting to note that the trend of the CPU load in Figure 17 for OLSR and the length of the data packets in Figure 13 is almost identical. This is due the increasing amount of CPU power necessary to process the embedded topology information as the amount of node pairs and therefore packet length increase.

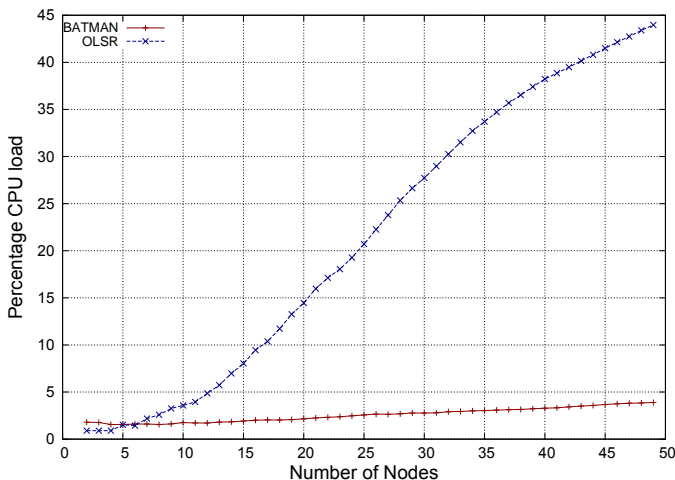


Fig. 17. Percentage of CPU used versus an increasing number of nodes in the network

The memory requirements of OLSR are also shown to increase at a far sharper rate than BATMAN as shown in Figure 18 due to its need to store complete routing tables for the whole network. BATMAN on the other hand only needs to store information about which of its local neighbours will be used to reach distant nodes. OLSR overtakes BATMAN in terms of memory requirements at 30 nodes but increases at a far sharper rate.

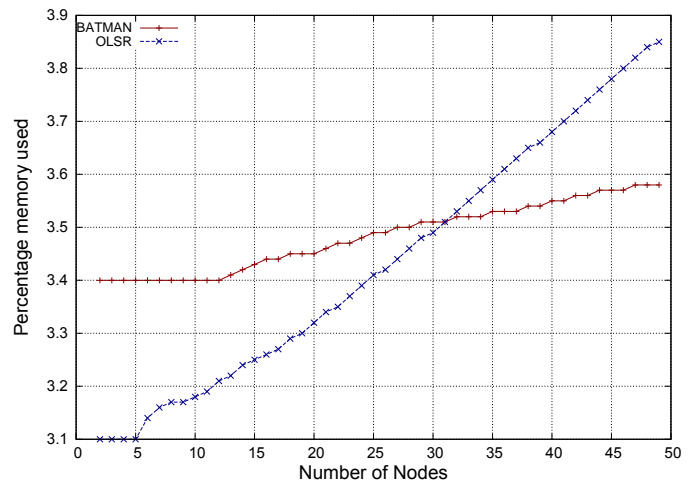


Fig. 18. Percentage of memory consumed versus an increasing number of nodes in the network

VI. CONCLUSION

The results from experiments done so far in the wireless grid lab with BATMAN and OLSR have shown that, for static wireless mesh networks, BATMAN outperforms OLSR on almost all performance metrics.

BATMAN's simple philosophy of not collecting more information than you can use and only getting information about your neighbours makes computation far more efficient. What is very encouraging is that a simplified protocol which exhibits a 10 fold improvement in CPU load at a network size of 49 nodes still shows a 17% improvement in throughput on average for any node pair in the network. However, OLSR did show a slight 1% advantage in packet loss as well as 0.5% advantage in successfully established links on average. But these are too small to be of any statistical relevance.

Many of the links in the wireless grid proved not to be symmetrical and BATMAN took full advantage of its ability to use non symmetrical links between nodes. Only 28% of its links were symmetrical compared to 61% for OLSR. BATMAN also proved its ability to stabilize on optimal routes and avoid a high degree of route flapping by only changing a route every 25 seconds per node as opposed to OLSR changing a route every 12 seconds per node on average.

BATMAN's Routing overhead is also significantly lower than OLSR in terms of number of bytes per second leaving a node. Results showed that BATMAN only used about 750 bytes per second of overhead as opposed to OLSR using 6000 bytes per second for the full 49 node network.

Both the low CPU load as well as the lower routing overhead also bode well for mesh networks that are trying to minimize power consumption when running on batteries being recharged using renewable energy sources.

These results demonstrate that new technical interventions often move from the primitive to the the complicated and back to the simple and BATMAN appears to be a watershed for MANET routing. Perhaps BATMAN is the panacea that

community wireless mesh networks have been waiting for, which will allow them to scale to large rural villages or across large cities on small low-cost low power wireless routers.

VII. FUTURE CONSIDERATIONS

Routing protocols are constantly evolving and this holds true for both BATMAN and OLSR. There is currently work being done on many fronts. The development community has launched work on OLSR - Next Generation (OLSR-NG) which seeks to allow OLSR to scale better. The aim is to allow OLSR to scale up to 10000 nodes with up to 20000 routes on embedded hardware with 200 MHz RISC CPU's and 16 MB of RAM. BATMAN is a parallel approach by many of the same developers to try something completely new

Within the IETF MANET a new version of OLSR was released by the academic community at INRIA called OLSRv2. OLSRv2 is however simply a small tweak of OLSR, it retains the same basic mechanisms and algorithms, while providing a more flexible signaling framework and some simplification of the messages being exchanged. It can also accommodate either IPv4 or IPv6 addresses in a compact manner.

What remains to be seen is which one of these three parallel activities achieves worldwide acceptance and in the end performs the most optimally for a wireless mesh network. BATMAN will soon be submitted as an Internet-Draft to the IETF MANET working group and it is hoped that it will begin to make more of the ad-hoc network community aware of the advantages of keeping routing protocols as simple as possible.

REFERENCES

- [1] P. Jacquet, P. Muhlethaler, T. Clausen, A. Laouiti, A. Qayyum, and L. Viennot, "Optimized link state routing protocol for ad hoc networks," *Multi Topic Conference. IEEE INMIC 2001. Technology for the 21st Century. Proc. IEEE International*, pp. 62–68, 2001.
- [2] D.B. Johnson and D.A. Maltz, "Dynamic source routing in ad hoc wireless networks," *Mobile Computing*, vol. 353, pp. 153–181, 1996.
- [3] C.E. Perkins and E.M. Royer, "Ad-hoc on-demand distance vector routing," *Proceedings of the 2nd IEEE Workshop on Mobile Computing Systems and Applications*, vol. 2, pp. 90–100, 1999.
- [4] J. Bicket, D. Aguayo, S. Biswas, and R. Morris, "Architecture and evaluation of an unplanned 802.11 b mesh network," *Proc. of the 11th Annual International Conference on Mobile Computing and Networking*, pp. 31–42, 2005.
- [5] K. Ramachandran, M. Buddhikot, G. Chandranmenon, S. Miller, E. Belding-Royer, and K. Almeroth, "On the Design and Implementation of Infrastructure Mesh Networks," *Proceedings of the IEEE Workshop on Wireless Mesh Networks (WiMesh)*, pp. 4–15.
- [6] G. Di Caro, F. Ducatelle, and L.M. Gambardella, "AntHocNet: an ant-based hybrid routing algorithm for mobile ad hoc networks," *Proceedings of Parallel Problem Solving from Nature (PPSN VIII)*, vol. 3242, pp. 461–470.
- [7] "Nsf workshop on network research testbeds, chicago, il," October 2002, http://wwwnet.cs.umass.edu/testbed_workshop/.
- [8] T.R. Andel and A. Yasinac, "On the credibility of manet simulations," *Computer*, vol. 39, no. 7, pp. 48–54, July 2006.
- [9] P. Gupta and P.R. Kumar, "The capacity of wireless networks," *IEEE Transactions on Information Theory*, vol. 46, no. 2, pp. 288–404, March 2000.
- [10] P. Gupta and R. Drag, "An experimental scaling law for ad-hoc networks," Tech. Rep., Bell Laboratories, 2006.
- [11] "IETF Mobile Ad-Hoc Networks (MANET) Working Group," 1 August 2007, <http://www.ietf.org/html.charters/manet-charter.html>.
- [12] IEEE, *Draft Standard for Information Technology - Telecommunications and Information Exchange Between Systems - LAN/MAN Specific Requirements - Part 11: Wireless Medium Access Control (MAC) and physical layer (PHY) specifications: Amendment: ESS Mesh Networking*, IEEE, New York, NY, USA, March 2007, P802.11s/D1.02.
- [13] D.S.J.D. Couto, D. Aguayo, J. Bicket, and R. Morris, "a high-throughput path metric for multi-hop wireless routing," *Wireless Networks*, vol. 11, no. 4, pp. 419–434, 2005.
- [14] A. Tonnesen, "Implementing and extending the optimized link state routing protocol," M.S. thesis, University of Oslo, Norway, 2004.
- [15] D.L. Johnson and A. Lysko, "Comparison of MANET Routing Protocols Using a Scaled Indoor Wireless Grid," *Mobile Networks and Applications*, 2008.
- [16] H. Lundgren, E. Nordström, and C. Tschudin, "Coping with communication gray zones in IEEE 802.11 b based ad hoc networks," *Proceedings of the 5th ACM International Workshop on Wireless Mobile Multimedia. WOWMOM 2002*, pp. 49–55, 2002.
- [17] K. Xu, M. Gerla, and S. Bae, "Effectiveness of RTS/CTS handshake in IEEE 802.11 based ad hoc networks," *Ad Hoc Network Journal*, vol. 1, no. 1, pp. 107–123, 2003.
- [18] K. Ramachandran, I. Sheriff, E. Belding-Royer, and K. Almeroth, "Routing stability in static wireless mesh networks," *Passive and Active Network Measurement*, vol. 4427, pp. 73–82, June 2007.
- [19] D.L. Johnson and G.P. Hancke, "Comparison of two routing metrics in OLSR on a grid based mesh network," *Ad hoc Networks*, 2008.