WPR: A Proactive Routing Protocol Tailored to Wireless Mesh Networks

Miguel Elias M. Campista, Luís Henrique M. K. Costa, and Otto Carlos M. B. Duarte
UFRJ - PEE/COPPE/GTA - DEL/POLI
E-mails:{miguel,luish,otto}@gta.ufrj.br

Abstract— **This work proposes the Wireless-mesh-network Proactive Routing (WPR) protocol for wireless mesh networks. Unlike current routing protocols, such as the Optimized Link-State Routing (OLSR), WPR uses a controlled-flooding algorithm tailored to the typical wireless-mesh-network traffic matrix, which concentrates traffic on links close to the gateway. The goal is to improve efficiency by saving network resources and avoiding network bottlenecks. WPR also avoids redundant messages using the AMPR (Adapted MultiPoint Relay) set. In this paper, we provide a complexity analysis of the algorithms used by WPR and OLSR. Besides, simulation results show that WPR outperforms OLSR in throughput and packet delivery rate.**

Keywords: Wireless mesh networks and routing protocol.

I. Introduction

Wireless mesh networks (WMNs) are a cost-effective solution for access networks given their low cost and relative ease of deployment. These networks improve connectivity and provide backhaul access to nodes not within range of gateways to the wired infrastructure. WMNs deploy a backbone of stationary wireless routers which communicate via multiple hops. Thus, routing play a key role in WMNs.

Most of wireless networking advantages are related to the broadcast nature of radiofrequency transmissions. Nevertheless, radiofrequency transmissions also experience fast link-quality changes, high medium attenuation, and interference. These characteristics lead to limited bandwidth if compared to wired technologies. Besides, protocols aiming at improving communication reliability generally introduce overhead, e.g. to acknowledge transmissions or to reproduce as closely as possible the current medium conditions. In WMNs, control overhead is especially undesirable because traffic converges towards wired-network gateways, possibly introducing bottlenecks. Therefore, it is important to save network resources to avoid congested areas.

At the routing layer, WMN protocols are designed to avoid overhead. Protocols such as Link Quality Source Routing (LQSR [1]) and Srcr [2] are fundamentally link-state based, but use route discovery procedures to only update the routing metrics of the commonly used links. They assume that frequently flooding the network with routing messages is not efficient because most communications include wired-network gateways. Proactive protocols, on the other hand, use control flooding techniques to reduce overhead. The Optimized Link-State Routing (OLSR) protocol defines a subset of neighbors in charge of forwarding routing messages. This subset is called MPR (MultiPoint Relay), which is composed of the minimum set of one-hop neighbors needed to reach all two-hop neighbors of the same node. The Fisheye State Routing (FSR [3]) protocol adjusts the time-to-live (TTL) field of each routing message to concentrate routing updates in the vicinity of the sender. The authors argue that most communications are between nearby nodes and therefore it is not worthy always flooding the whole network.

In this work, we propose a proactive link-state-based routing protocol called Wireless-mesh-network Proactive Routing (WPR). Using a proactive approach, we avoid the initial latency of route discovery procedures. WPR also introduces a controlled-flooding algorithm tailored to the typical traffic matrix of wireless meshes. Unlike OLSR and FSR, which reduce overhead in an expanding-ring fashion, WPR updates more frequently the route metrics of the most-used links to reach the wired-network gateway and vice-versa. To accomplish this, WPR approximates the network topology to a tree, where the wired-network gateway plays the role of root. WPR introduces the AMPR (Adapted MultiPoint Relay) set, which reduces redundant messages as the MPR set of OLSR. This set is computed according to the controlled-flooding algorithm. We compare the complexity of the proposed algorithms with their OLSR counterparts. Besides, we analyze the performance of WPR via simulations with ns-2. Results show that in presence of backbone internal traffic, the aggregated throughput with WPR is up to 24% better than with OLSR.

This paper is organized as follows. Section II discusses our work assumptions. Section III introduces the WPR protocol. The complexity of the algorithms used by WPR are analyzed in Section IV. Section V describes the simulation setup and the modifications needed to the ns-2 simulator. Simulation results are presented in Section VI. Finally, Section VII concludes this paper and identifies possible future directions.

II. Work Assumptions

In this work, we first assume that many wireless-mesh-network applications require Internet access [1]. Nevertheless, applications where the source-destination pair resides inside the wireless mesh are also possible (e.g. P2P applications). In this case, although most traffic is among peers, access to servers in the Internet is often needed. Our second assumption is that WPR nodes have complete knowledge of the network topology. This requirement is achieved by using a proactive link-state protocol. Our third assumption is that nodes know the IP address of the wired-network gateways to control flooding. That information can be easily obtained once the gateways announce their default external route to the wireless network. Finally, we assume that the routing protocol is run only on backbone nodes. If a user is willing to connect the network, he must use a backbone node as an access point.

III. WPR PROTOCOL

Link-state protocols require nodes to be aware of the complete topology. Hence, nodes periodically flood the network with link states. In wireless networks, this implies in multiple accesses to the shared medium. Thus, most controlled-flooding protocols aim at reducing the transmission frequency of link-state updates from seldom-used or redundant links [4].

In this work, we propose a proactive link-state-based protocol called WPR (Wireless-mesh-network Proactive Routing). WPR has two main features: (i) a controlled-flooding algorithm and (ii) an algorithm to compute the AMPR (Adapted MultiPoint Relay) set. The controlled-flooding algorithm aims at reducing the overhead in wireless mesh networks by taking into account their typical traffic matrix. In addition, the WPR protocol uses an algorithm to compute the AMPR set which avoids redundant routing messages. The AMPR set function is similar to the MPR set of OLSR. The AMPR set, however, is computed according to our proposed controlled-flooding algorithm. The routing messages structure and the protocol operation are similar to those used by OLSR.

The controlled-flooding algorithm of WPR considers that communications converge towards gateways. Hence, the topology is approximated to a tree, whose root is the gateway. Based on the tree topology, the controlled-flooding algorithm of WPR identifies as the most-used links by a node i, those connecting nodes in two different sets: the ascendent and the descendent sets. Let $\mathcal{A}_i$ denote the ascendent set of i and $\mathcal{D}_i$ the descendent set. A node a_1 is in the $\mathcal{A}_i$ set if it is on the path from i to the gateway (g), whereas a node d_1 is in $\mathcal{D}_i$ if node's d_1 path to the gateway includes node i. Figure 1 depicts both sets, where $\mathcal{A}_i = \{a_1, a_2, ..., g\}$ and $\mathcal{D}_i = \{d_1, d_2, ...\}$.

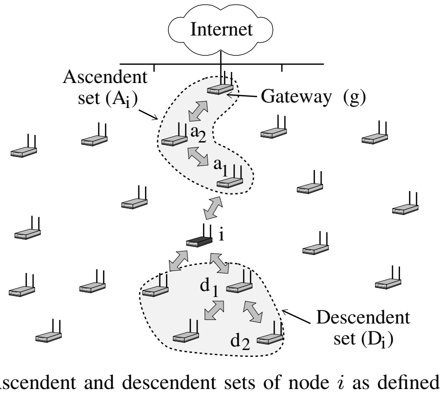


Fig. 1. Ascendent and descendent sets of node i as defined by WPR.

The controlled-flooding algorithm of WPR reduces routing overhead by limiting the number of nodes that forward routing messages from node i. In WPR, the forwarding nodes are those within the $\mathcal{A}_i$ or the $\mathcal{D}_i$ sets. Therefore, a routing message from i is only forwarded by nodes in $\mathcal{A}_i$, because i is a descendent; and by nodes in $\mathcal{D}_i$, because i is an ascendent of those nodes. Forwarding routing messages from ascendent nodes allows descendents to compute routes to the gateway, whereas forwarding routing messages from ascendents allows descendents to compute routes in the reverse direction. Figure 2 illustrates the controlled-flooding algorithms used by OLSR and WPR. In this figure, the external dotted line represents the range of the OLSR flooding algorithm. In this case, even using the MPR set, all nodes receive a route update from i. The internal dotted line, on the other hand, shows the range of a controlled-flooding message sent by node i using WPR. Node i forwards control messages from nodes in $\mathcal{T}_i$ set, where $\mathcal{T}_i = \mathcal{A}_i \cup \mathcal{D}_i$, reducing the routing overhead.

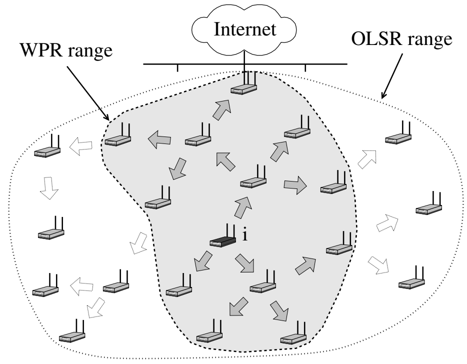


Fig. 2. Controlled-flooding algorithms.

The $\mathcal{A}_i$ set is known with routing computation. On the other hand, the $\mathcal{D}_i$ set is known based on HELLO messages. In these messages, every node lists its neighbors and indicates if the neighbor was chosen as one-hop ascendent. Thus, node i knows its one-hop descendents and can forward their messages. To forward routing messages originated at any node in $\mathcal{D}_i$, a recursive property is used. If a routing message was at last forwarded by a one-hop descendent of node i, e.g. d_1, it is assumed that one of the d_1's descendents has originated the message. As d_1's descendents are also in $\mathcal{D}_i$, we have a recursive property which is a simple solution to identify descendent nodes.

WPR uses topology-control messages to disseminate link states, similarly to OLSR. Nevertheless, a flag is needed in the message header to indicate the dissemination type. In WPR, in addition to controlled-flooding messages, the whole network is also periodically flooded to guarantee that all nodes will be aware of the complete network topology. We maintain network-wide flooding because communications not including gateways are also possible. As a consequence, WPR defines a period T to send control messages and the number of controlled-flooding messages per flooding messages.

Algorithm 1 shows the operation of the controlled-flooding algorithm of WPR, describing the action taken by node i upon receiving a control message μ_c. In the algorithm, we denote s as the source node of the control message μ_c, v the last neighbor to forward μ_c, g the wired-network gateway, and $\mathcal{M}_v$ the set of AMPR nodes of v. A node only forwards a routing message if it is in the AMPR set of the last forwarding node v. If this condition is true, node i examines the type of control message received as identified on the μ_c header. If μ_c is a flooding message, node i simply forwards it. Otherwise, if μ_c is a controlled-flooding message, node i uses two functions to verify whether node v is in its $\mathcal{T}_i$ set. Function $\texttt{ascendent}(s, i, g)$ verifies if s is in the $\mathcal{A}_i$ set of node i, with respect to gateway g. On the other hand, function $\texttt{descendent}(v, i, g)$ verifies if node v is a one-hop descendent of node i. Node i does not forward μ_c if s and v are neither in $\mathcal{A}_i$ nor in $\mathcal{D}_i$.

Algorithm 1 CONTROLLED-FLOODING ALGORITHM.

```
if i ∈ M_v then
   if μ_c is a flooding message then
      forward(μ_c);
   else if μ_c is a controlled-flooding message then
      if ascendent(s, i, g) or descendent(v, i, g) then
         forward(μ_c);
      else
         discard(μ_c);
      end if
   end if
end if
```

The relationship between the transmission of controlled-flooding and flooding messages is adjusted according to the distance of each node to the gateway, in number of hops. The closer the node to the gateway, the higher the number of descendents. Therefore, nodes near the gateway are already in a higher number of $\mathcal{T}_i$ sets, and there is no need for them to flood the network at the same frequency as farther nodes. Each WPR node then adjusts the relation between controlled-flooding and flooding messages according to $r(h) = f(n) - h$, where $r(h)$ is the number of controlled-flooding messages per flooding message, h is the number of hops to reach the gateway, and $f(n)$ is the maximum number of controlled-flooding messages per flooding message computed as a function of the number of nodes (n) in the network. The value of n is obtained from the known network topology. Function $f(n)$ is equal to $R_{min} + \sqrt{n}$, where R_{min} is the minimum number of controlled-flooding messages sent per flooding message. In our simulations, we use a grid scenario. Thus, it is convenient in this work to use $r(h)$ as a function of the square root of the number of nodes.

Algorithm 2 AMPR SET COMPUTATION ALGORITHM.

```
if V_{2,i} ≠ ∅ then
   M_i ⇐ compute_AMPR((V_{1,i} ∩ T_i),(V_{2,i} ∩ T_i));
   if M_i = ∅ then
      M_i ⇐ compute_AMPR(V_{1,i},V_{2,i});
   else
      V'_{2,i} ⇐ exclude_nodes(M_i,V_{2,i});
      if V'_{2,i} ≠ ∅ then
         M_i ⇐ M_i ∪ compute_AMPR((overline{V_{1,i} ∩ T_i}),V'_{2,i});
      end if
   end if
end if
```

The second fundamental characteristic of WPR is the utilization of the AMPR (Adapted MPR) set to avoid redundant routing messages. The key idea is to find the one-hop neighbors within $\mathcal{T}_i$ needed to reach all two-hop neighbors in $\mathcal{T}_i$. As WPR also floods the network, the AMPR set is complemented with the one-hop neighbors not in $\mathcal{T}_i$ needed to reach all two-hop neighbors also not in $\mathcal{T}_i$. The algorithm to compute the AMPR set is therefore run twice (Algorithm 2). Let $\mathcal{M}_i$ denote the AMPR set of node i, $\mathcal{V}_{1,i}$ the set of one-hop neighbors of i, and $\mathcal{V}_{2,i}$ the set of two-hop neighbors of i. If $\mathcal{V}_{2,i}$ is not empty, the algorithm first computes the subset of nodes in $\mathcal{M}_i$ in charge of sending controlled-flooding messages to two-hop neighbors also in $\mathcal{T}_i$. Thus, the subset of nodes in $\mathcal{V}_{1,i}$ and in $\mathcal{V}_{2,i}$ that are also in $\mathcal{T}_i$ are passed as parameters to function compute_AMPR. It is worth mentioning that $\mathcal{V}_{2,i}$ are obtained from HELLO messages sent by nodes in $\mathcal{V}_{1,i}$. If the AMPR set remains empty after function compute_AMPR is run, the computation of the AMPR set becomes equivalent to compute the MPR set. This is because the parameters passed to function compute_AMPR are $\mathcal{V}_{1,i}$ and $\mathcal{V}_{2,i}$, resulting in a set of one-hop neighbors able to reach all two-hop ones. If $\mathcal{M}_i$ is not empty, on the other hand, the algorithm excludes nodes in $\mathcal{V}_{2,i}$ that are already reached by nodes in $\mathcal{M}_i$. This results in $\mathcal{V}'_{2,i}$ set, which is composed of two-hop nodes of i that are still not reached by nodes in $\mathcal{M}_i$. The algorithm runs again function compute_AMPR to complement $\mathcal{M}_i$ with the one-hop neighbors needed to reach all nodes in $\mathcal{V}'_{2,i}$. The one-hop neighbors used at this time are not in $\mathcal{T}_i$, since the one-hop neighbors in $\mathcal{T}_i$ were already used. The second time function compute_AMPR is run guarantees that all two-hop nodes are reached when node i floods the network. The final AMPR set is the union of the nodes found in the first and in the second time function compute_AMPR is run. Note that the one-hop ascendent of i must be always in $\mathcal{M}_i$. Otherwise, a routing message from a node i would never be forwarded to all ascendent nodes.

The proposed controlled-flooding algorithm and the AMPR set reduce overhead, resulting in bandwidth availability.

IV. COMPLEXITY ANALYSIS

In this section, we analyze the complexity of WPR and OLSR in terms of AMPR and MPR sets computation, and number of routing messages sent. We use the same notation of Section III.

Lemma 1: Let $|\mathcal{V}_{1,i}|$ and $|\mathcal{V}_{2,i}|$ denote respectively the number of one-hop and two-hop neighbors of node i, and $v = |\mathcal{V}_{1,i}| = |\mathcal{V}_{2,i}|$. In the worst case, i.e. where every one-hop neighbor is in the MPR set, the complexity to an OLSR node compute its MPR set is $O(v^3)$.

Proof: In OLSR, each node i needs to be aware of its one-hop and two-hop neighbors to compute its MPR set. Choosing a node as an MPR requires node i to find its one-hop neighbor with the highest adjacency degree, considering only two-hop neighbors. Node i then counts the number of adjacencies each one-hop neighbor has comparing to all two-hop ones. This operation takes $O(v^2)$ steps, where each step is an adjacency test. As soon as an MPR node is chosen, it is removed from the one-hop neighbor list. In addition, the two-hop neighbors reached by the new MPR are removed from the two-hop neighbor list. Selecting a node as an MPR is a recursive procedure. Thus, it is repeated until all two-hop neighbors are reached by at least by one MPR node. In the worst case, computing the MPR set requires $\sum_{\xi=1}^{v} \xi^2$ steps, increasing the complexity to $O(v^3)$. ■

Lemma 2: Let $v_{in} = |\mathcal{V}_{1,i} \cap \mathcal{T}_i| = |\mathcal{V}_{2,i} \cap \mathcal{T}_i|$ denote the number of one-hop and two-hop neighbors in $\mathcal{T}_i$, and $v_{out} = |\overline{\mathcal{V}_{1,i} \cap \mathcal{T}_i}| = |\mathcal{V}_{2,i}^{'}|$ the number of one-hop and the remaining two-hop neighbors *not* in $\mathcal{T}_i$. Thus, the complexity for a WPR node compute its AMPR set is $O(v_{in}^3 + v_{out}^3)$.

Proof: In WPR, the AMPR set computation is similar to computing the MPR set twice. Nevertheless, the set of one-hop and two-hop neighbors used in both runs are subsets of the one-hop and two-hop neighbor sets used by OLSR. Computing the nodes in the AMPR set in charge of forwarding controlled-flooding messages requires $\sum_{\xi=1}^{v_{in}} \xi^2$ steps, therefore it is $O(v_{in}^3)$. Similarly, computing AMPR nodes to reach nodes *not* in $\mathcal{T}_i$ is $O(v_{out}^3)$. Hence, computing the complete AMPR set is $O(v_{in}^3 + v_{out}^3)$. ■

Theorem 1: The complexity to compute the AMPR set is less than or equal to compute the MPR set.

Proof: In Lemma 1, the number of one-hop and two-hop neighbors of i is the same and equal to v. On the other hand, in Lemma 2, $v_{in} + v_{out} \leq v$. Therefore, the number of steps taken to compute the MPR set by an OLSR node is greater than or equal to the number of steps a WPR node requires to compute the AMPR set. This is because $v_{in}^3 + v_{out}^3 < v^3$, unless $v_{out} = 0$ (forwarding-chain topology). ■

Lemma 3: Let n denote the number of nodes in the network. The number of controlled-flooding messages sent using OLSR is $O(n^2)$.

Proof: Considering again that the MPR set of a node i is equal to its one-hop neighbor set, the number of times a routing message from i is forwarded in the network is n. As all n nodes periodically flood the network, the number of controlled-flooding messages is $O(n^2)$. ■

Lemma 4: The number of controlled-flooding messages sent using WPR is $O((\sum_{i=1}^{n}(|\mathcal{A}_i| + |\mathcal{D}_i|)/n)^2)$.

Proof: Let l_i denote the level of node i in the tree topology represented by $\mathcal{T}_i$, and $|\mathcal{A}_i|$ and $|\mathcal{D}_i|$ the number of nodes in $\mathcal{A}_i$ and $\mathcal{D}_i$, respectively. Thus, the number of times a controlled-flooding message from i is forwarded is $\sum_{l=0}^{l_i-1} |\mathcal{A}_i| + \sum_{l=l_i}^{l_{max}} |\mathcal{D}_i| = l_i + \sum_{l=l_i}^{l_{max}} |\mathcal{D}_i|$. Note that $|\mathcal{A}_i|$ and $|\mathcal{D}_i|$ depend on the level of i in the tree. Considering that there are n nodes in the network, the average number of ascendents and descendents in the network is $\sum_{i=1}^{n} |\mathcal{A}_i|/n$ and $\sum_{i=1}^{n} |\mathcal{D}_i|/n$, respectively. Thus, the number of controlled-flooding messages is $O((\sum_{i=1}^{n}(|\mathcal{A}_i| + |\mathcal{D}_i|)/n)^2)$. ■

Theorem 2: The WPR protocol introduces less overhead compared with OLSR.

Proof: The number of controlled-flooding messages is lower when using WPR if $(\sum_{i=1}^{n}(|\mathcal{A}_i| + |\mathcal{D}_i|)/n) < n$, according to Lemmas 3 and 4. Thus, WPR introduces less overhead than OLSR if for at least one node i, $|\mathcal{A}_i| + |\mathcal{D}_i| < n$. This would not be true only in forwarding-chain topologies where $|\mathcal{A}_i| + |\mathcal{D}_i| = n$. ■

V. Simulation Setup

WPR is analyzed via simulations using ns-2.31. Nevertheless, the PHY-layer and the IEEE 802.11 MAC-layer modules available are rather simplistic. Hence, we present our improvements in the simulator code to make our results more realistic, and also some WPR implementation issues. In addition, we describe the scenario and the traffic pattern used.

A. Physical layer

Originally, ns-2.31 does not consider bit error rate (BER) in wireless transmissions. The PHY-layer module computes the maximum interference and reception ranges according to pre-determined values of transmission and reception power. In this work, the PHY-layer module developed in [5] was adapted to ns-2.31. In this module, the BER is computed as a function of the channel signal-to-noise-ratio and the physical transmission rates defined by IEEE 802.11b. A packet loss probability is associated depending on the BER computed per frame. The module is based on experimental results, thus it considers possible variations on medium conditions. We use the shadowing model to compute transmission ranges and set parameters according to experimental results suggested in [6].

B. MAC layer

We implement the AutoRate Fallback (ARF [7]) algorithm in the MAC-layer module of ns-2.31. This algorithm is simple and its operation is based on two counters and a timer. In this work, we use a 60 s timer [7].

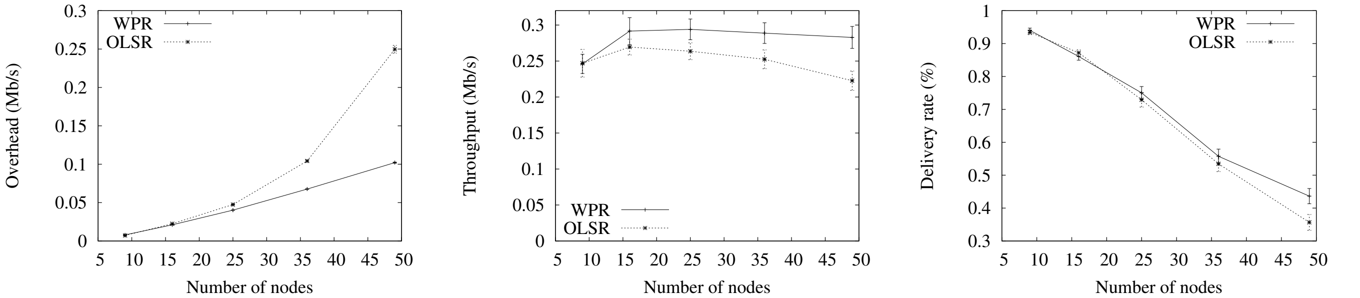
C. Routing layer

We adapt the OLSR module to the simulator version used [8] and we implement the metric ETX in OLSR according to its real implementation in the OLSR daemon [9]. We use a link quality window size of 20 s.

In this work, OLSR *always* uses the ETX metric. We use our adapted version of OLSR in ns-2.31 as a basis for WPR implementation. In our simulations, both WPR and OLSR periodically send route updates at intervals (T) of 5 s, as suggested in RFC 3626 of OLSR. The minimum relation used by WPR between controlled-flooding and flooding messages (R_{min}) is set to 13. This relation is also used by the Fisheye extension of OLSR and is adjusted according to the number of hops needed to reach the gateway (Section III).

D. Traffic pattern and simulated scenario

We use two different traffic patterns in our simulations [10]. We call them web-like and internal traffic. The web-like traffic models requests for Internet web pages which generate multiple responses containing one object each. The size of the request is constant and equal to 1 kbyte. In addition, the interval of time between different pages and between the transmission of consecutive objects is represented by a random variable exponentially distributed with average equal to 10 s. The size of each object is modeled according to a random



(a) Overhead using web-like and internal traffic. (b) Throughput using web-like and internal traffic. (c) Delivery rate using web-like and internal traffic.

Fig. 3. Simulation results.

variable with Paretto II distribution with shape 1.2 and average of 12 kbytes. The web-like traffic uses TCP and is always bidirectional, where the request for a web page is always initiated by a backbone node to a wired node. We assume the bottleneck is the wireless part, therefore we do not simulate the influence of the wired network. The internal traffic models communications between nodes within the wireless mesh. The traffic is modeled using constant bit rate (CBR) sources over UDP. The duration of each flow is computed according to a random variable exponentially distributed with average of 48 s. The CBR communicating pairs as well as the web-like sources are randomly chosen among all network nodes but the gateway. The duration of each simulation run is 90 s.

Nodes are positioned in a grid, where the gateway is at one vertex. The size of the network ranges from 9 to 49 nodes separated by 20 m. Thus, each node has at most 20 neighbors at 1 Mb/s and 8 at 11 Mb/s according to our PHY-layer parameters. We use one gateway.

VI. Simulation Results

The performance of WPR is compared with OLSR according to three metrics: routing overhead, network aggregated throughput, and delivery rate at the application level. All results are obtained using a confidence level of 95%.

In our analysis, we combine web-like and internal traffic. As we concentrate route updates on links towards the gateway, the topology maps known by the different nodes may not be synchronized. This can affect communications between nodes within the wireless mesh. In this test, 25% of the backbone nodes are CBR sources transmitting data at 56 kb/s, and other 25% of the backbone nodes produce web-like traffic using 20 sessions. Each session is a sequence of request and reception of web-page objects. Figure 3(a) plots the overhead introduced by OLSR and WPR. We define routing overhead as the number of routing messages transmitted in the network. Observe that WPR reduces the routing overhead by up to 60%, which complements our complexity analysis. Figure 3(b) shows the aggregated throughput obtained by WPR and OLSR. It is observed that reducing the routing overhead has a direct impact on the network aggregated throughput. As mentioned earlier, in wireless mesh networks, reducing overhead can avoid network bottlenecks on links close to the gateway. The aggregated throughput of WPR is approximately 24% better than OLSR. Figure 3(c) shows that the delivery rate at the application level of WPR is up to 22% better than with OLSR. This also reflects the gains with the reduction of routing overhead.

VII. Conclusion

This work proposes WPR, a link-state routing protocol for wireless mesh networks. WPR is characterized by the AMPR (Adapted MultiPoint Relay) set and by the controlled-flooding algorithm tailored to wireless meshes. Our complexity analysis has shown that the AMPR set computation requires fewer steps than the computation of the MPR (MultiPoint Relay) set used by the Optimized Link-State Routing (OLSR) protocol. In addition, we have shown that WPR introduces less controlled-flooding messages than OLSR. The performance of WPR was also evaluated via simulations. Our results complemented our complexity analysis, and confirmed the better performance of WPR despite our assumption on traffic convergence towards gateways. Our future work includes extending our simulation analysis and implementing WPR in a real prototype.

Acknowledgments

This work was supported by CNPq, FAPERJ, and FUJB.

References

[1] R. Draves, J. Padhye, and B. Zill, "Routing in multi-radio, multi-hop wireless mesh networks," in *ACM MOBICOM*, Sept. 2004, pp. 114–128.
[2] J. Bicket, D. Aguayo, S. Biswas, and R. Morris, "Architecture and evaluation of an unplanned 802.11b mesh network," in *ACM MOBICOM*, Aug. 2005, pp. 31–42.
[3] G. Pei, M. Gerla, and T.-W. Chen, "Fisheye state routing in mobile ad hoc networks," in *ICDCS Workshop on Wireless Networks and Mobile Computing*, Apr. 2000, pp. D71–D78.
[4] 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.
[5] W. Xiuchao and A. L. Ananda, "Link characteristics estimation for IEEE 802.11 DCF based WLAN," in *IEEE LCN*, Nov. 2004, pp. 302–309.
[6] W. Xiuchao, "Simulate 802.11b channel within NS2," National University of Singapore, Tech. Rep., 2004.
[7] G. Holland, N. H. Vaidya, and P. Bahl, "A rate-adaptive MAC protocol for multi-hop wireless networks," in *ACM MOBICOM*, July 2001, pp. 236–251.
[8] F. J. Ros, Acessed in http://masimum.dif.um.es/um-olsr/html/, 2005.
[9] olsrd [on-line], Accessed in http://www.olsr.org, 2007.
[10] R. Baumann, S. Heimlicher, V. Lenders, and M. May, "HEAT: Scalable routing in wireless mesh networks using temperature fields," in *IEEE WoWMoM*, June 2007, pp. 1–9.