An Overlay Maintenance Protocol for Overlay Routing on Top of Ad Hoc Networks

Sandrine Calomme and Guy Leduc

Research Unit in Networking
Electrical Engineering and Computer Science Department
University of Liège, Belgium

Abstract. The protocol described in this paper builds and maintains an overlay topology on top of an ad hoc network. The overlay is intended to be used by a routing application. As flooding is a key component of many route discovery mechanisms in MANETs, we evaluate the delivery percentage, bandwidth consumption and time duration of flooding a message on the overlay. We also consider the overlay path stretch as an indicator for the data transfer transmission time.

The protocol does not require any information from the underlay routing protocol, nor cooperation from the nodes that do not belong to the overlay. Each overlay node maintains a set of nearest overlay nodes and exchanges its neighbourhood information with them in order to select useful overlay links. Resilience is afforded by setting a minimum number of overlay neighbours.

The performance observed over OLSR are good, for all overlay densities and mobility level studied.

1 Introduction

The overlay technique consists in building a virtual topology over the network physical topology of a network. At the transport level, the overlay links are tunnels that can be laid over several physical links. Layer 4 overlays have been successfully proposed in the Internet for improving the basic routing service provided by IP for a variety of applications. They can afford for example a multicast functionality, object location, secure data dissemination, content-distribution, quality of service or fault-tolerant routing. They have also been successful for in-situ testing of new networking solutions, a field that recently attracted a growing interest [1].

Many overlay applications have been proposed, and many with their own proprietary functionality support. Several overlay functionalities are covered by different works. These are for example the overlay topology discovery, routing path selection, fault detection and tolerance, overlay link performance estimation or resource allocation. Hence, the idea of designing a general unified framework for various application-specific overlays emerged. A generic overlay solution allows the designer of an overlay application to concentrate on its application goal and needs rather than on the maintenance of the overlay [2, 3, 4].

There are several incentives for deploying overlays on MANETs. Firstly, the ad hoc users are likely to desire the same services as the one offered when they are connected through wired or infrastructure mode wireless access to the Internet. Secondly, ad hoc networks and overlay applications are faced with the same fundamental challenge of providing connectivity in a decentralised, dynamic environment. In both domains, the user devices work simultaneously as end-system and routing nodes. This is an advantage compared to the Internet, where core routers that take an active role in an overlay cannot benefit from useful information own by the user application, and where end-host based overlays suffer from performance limitations. However, the maintenance of an overlay on top of an ad hoc network is challenging because of mobility and low bandwidth.

In a wired context, a complete decoupling of overlay topology and underlying data network topology may not be completely desirable because it leads to inefficiencies. In ad hoc networks, the maintenance of many long tunnels is not even conceivable. Mobility makes long routes very unreliable. The cost of rebuilding long tunnels may be important, in terms of delay and of bandwidth. Hence, the benefits of the overlay application could be completely lost if overlay neighbours were not close to each other.

We thus present in this paper a protocol that is able to build and maintain a locality-aware overlay on top of an ad hoc network. We do not specify its application domain and evaluate the virtual structure obtained by an indirect performance evaluation based on overlay flooding. The diffusion of message can be seen as a worst-case scenario for group communication. It is also a key component of many unicast route discovery mechanisms in MANETs.

2 Related Work

The deployment of layer 4 overlays in MANETs has not yet received a lot of attention. In [5], an overlay approach is proposed for service discovery in ad hoc networks. A new layer is inserted above the transport layer in order to build, maintain, and offer an overlay structure that optimally serves the mechanisms of service trading and efficiently adapts to the changing physical topology. The maintenance protocol consists of an adaptation of CAN that lightens its structure. It selects short overlay links and reduces the control traffic. The adaptation of application-layer multicast to the ad hoc environment was proposed by the AMRoute protocol [6], which establishes a mesh between the multicast users and then a tree for data distribution over the mesh. More proposals are available for P2P content distribution in MANETs. Many of them exploit cross-layer interactions with the routing protocol [7,8,9], or the node position knowledge [10] or even introduce the necessary p2p functions at the network routing layer [11,12]. An elegant p2p design for ad hoc networks is proposed in [13]. However, the authors do not take into account mobility as they are mostly interested in evaluating the feasibility of DHT lookup in ad-hoc networks.

In all above cited works, the use of logical long-range neighbours is avoided because of their prohibitive maintenance cost. Another common point is that the overlay quality is evaluated with the level of performance obtained by its user application.

3 The Overlay Topology Control Protocol

3.1 Protocol Overview and Assumptions

We denote our protocol by OTC, standing for Overlay Topology Control protocol. The algorithm is fully distributed and local, i.e. each overlay node exchanges only a few messages with a limited number of nearest overlay nodes. It builds connected overlay topologies, at least with a high probability.

We consider a connected underlay and assume that a routing protocol is available to all ad hoc nodes. The underlay routing protocol is supposed to provide short paths, but not necessarily the shortest ones and may build asymmetric paths. In order to work properly, the OTC protocol must obtain the distance between overlay nodes. We however do not assume that the underlay routing protocol is able to inform the above layer about the length of the available paths. We make instead the following, weaker, assumption: When a node receives a packet, it is able to know how many hops the packet has traversed since its emission.

Each overlay node U:

1. Collects in its neighbour candidate list L_U the identifier and the shortest distance to its K closest overlay nodes.
2. Also inserts in L_U any overlay node V such that $U \in L_V$ (thus turning the neighbourhood relation into a symmetric relation).
3. Selects its overlay neighbours in L_U by applying the pruning rule described below (in Sec. 3.5).
4. For resilience, does not prune:
   - its 3 nearest overlay nodes, nor
   - any overlay node located only at one hop.

For all simulations, we distributed the overlay nodes randomly and uniformly on the set of ad hoc nodes. The overlay density is defined as the proportion of overlay nodes. We observed in an extensive set of simulations that setting K to the value of 8 was sufficient to guarantee connectivity of overlay topologies with a probability higher than 95% for up to 1000 underlay nodes and overlay densities ranging from 10 to 90% [14].

Each overlay node must update its neighbour candidates list as soon as feasible when nodes move. Possible updates are adding, sorting and deleting elements. The overlay nodes must also determine if a neighbour candidate must be selected as neighbour or pruned.

We differentiate broadcast and unicast overlay neighbours. The former are also physical neighbours, i.e. there exists a direct radio communication link between them, while the distance between the latter is at least of two hops. We begin the OTC protocol description with the simple maintenance procedure of broadcast neighbours.

3.2 The Discovery and Maintenance of Broadcast Neighbours

Each overlay node regularly emits an OTC_HELLO message, encapsulated in a broadcast packet with the Time To Live (TTL) field set to one. If a node U receives an OTC_HELLO message from a node V, it adds V to its neighbour candidates list L_U. Every broadcast neighbour is automatically selected as overlay neighbour. Broadcast neighbours are purged if no hello message has been received during a given time interval.

If a node U has less than K broadcast overlay neighbours, its neighbour candidates list must be supplemented by overlay neighbours located further. The necessary unicast overlay neighbours will be selected among them. The rest of this section is devoted to this more complicated part of the OTC algorithm.

3.3 The Discovery of New Unicast Neighbour Candidates

As soon as a node enters the overlay, it regularly emits an OTC_REQUEST in broadcast packets. The Time To Live (TTL) field of these packets is set to increasing values, beginning from 2, until L_U gets sufficiently long (at least K neighbour candidates).

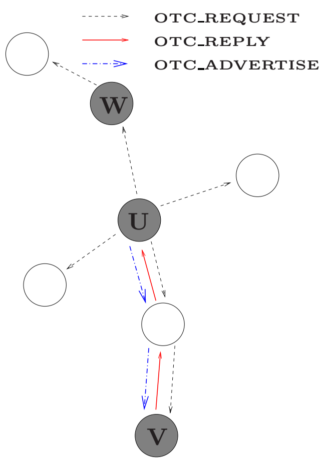
An overlay node V that receives an OTC_REQUEST from U responds with a unicast packet containing an OTC_REPLY if and only if node U is not already in L_V.

The neighbour candidates list is sorted by increasing distance and, when distances are equal, by increasing identifier. When node U receives the OTC_REPLY from node V, it calculates at which position it would insert V in L_U, using the number of hops the OTC_REPLY has passed through and V's identifier. If the position is less than or equal to K, it inserts V in its neighbour candidates list, sets its monitoring state for V on, and sends a unique OTC_ADVERTISE message to V. At this point of the protocol, this message is used by U for forcing V to create the (U, V) pair of neighbour candidates. This ensures the symmetry of the neighbourhood relationship.

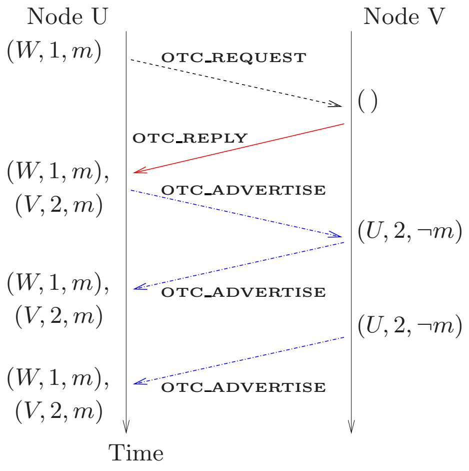
Node U maintains a monitoring state for each of its neighbour candidates. If node U monitors node V (i.e. its monitoring state for node V is on), this means that node U is responsible for estimating the distance $d(U, V)$, and communicating any change to V. The complete distance update process is described below, on Section 3.4.

If $U \notin L_V$ when node V receives the first advertisement from U, node V inserts U in its neighbour candidates list, and registers the distance $d(U, V)$ announced in the advertisement. It sets its monitoring state for U off and begins to send OTC_ADVERTISE messages to U at regular intervals. The reception of these frequent advertisements by node U makes it capable of monitoring the distance between U and V, by observing the number of hops they traversed. At stability, there is one and only one end node per candidate overlay link that monitors its length.

A simple discovery process is illustrated on Fig. 1. The overlay nodes – U, V and W – are grey-shaded. At the beginning, $L_U = ((W, 1, m))$, which means that U only knows one broadcast neighbour W and monitors it[1]. At the end, U and V have been inserted in the neighbour candidates list of each other, with distance $d(U,V) = 2$. Node U monitors node V, V does not monitor U (indicated in the Fig. 1(b) by the expression $\neg m$). Note that the distances registered in L_U and L_V are equal because it corresponds to the number of hops traversed by the reply, even if the request or the first advertisement has followed an underlay path of different length. The general rule is that the distance recorded for any overlay link equals the number of hops observed by the monitoring node at reception of an OTC message. In our example, these are the reply and subsequent advertisements flowing from node V to node U.



(a) OTC messages



(b) Corresponding evolution of the neighbours lists along time

Fig. 1. Discovery procedure

3.4 The Maintenance of the Neighbour Candidates List

Consider an overlay node U. In a mobile context, the content of its ordered neighbour candidates list, L_U, evolves continuously. The set of K closest overlay nodes of U and the distance between U and a member of L_U may change. Moreover, overlay node U may enter or leave the set of K closest overlay nodes of other overlay nodes.

New unicast overlay neighbour approaching.. Consider a node U that already knows at least K neighbour candidates. Let us define its current range R_U by the distance registered for the Kth element of L_U. If $R_U \leq 2$, interesting new neighbours are at most one hop away, and their discovery will be done through the reception of their hello messages. When $R_U > 2$, in order to spot

[1] The monitoring state for broadcast neighbours is not used by the algorithm and set on by default.

approaching unicast overlay nodes, node U regularly emits new OTC_REQUEST messages. The TTL field of the broadcast packet containing these requests is set to $R_U - 1$. As U owns a sufficient number of neighbour candidates, new overlay nodes located R_U hops or farther are not considered of better quality than the elements of L_U and must not be sought. The process induced by the reception of these requests is the same as described above.

Distance update. The updates are made possible by the regular emission of OTC_ADVERTISE messages. An overlay node U sets an advertisement timer for each of its neighbour candidates. The advertisement timer set for a node V is reset every time U sends an advertisement to V or receives an advertisement from V.

An overlay node assigns a longer expiration time to candidates it monitors than to candidates that it does not monitor. The difference of expiration time is such that, if there is no advertisement loss nor distance change, an overlay node only receives regular advertisements from neighbour candidates that it is monitoring. If it notices a distance modification, it sends a unique OTC_ADVERTISE to the corresponding, unmonitoring, peer. The overlay nodes thus also receive asynchronous advertisements from neighbour candidates that they are not monitoring, for being informed of any modification of the corresponding candidate overlay links.

If there are losses or if both nodes of a neighbour candidates pair are in the monitoring state advertisements are still received. The latter case may appear in transient scenarios, caused by mobility or at set up. For example when overlay nodes U and V discover the existence of each other in a short interval of time, by the reception of two OTC_REQUEST messages sent in opposite direction.

When an overlay node U receives an advertisement from V on a path of h hops, it first checks its monitoring state for V and updates $d(U, V)$:

- If node U is not monitoring V, it sets its local $d(U, V)$ variable to the value indicated in L_V.
- If node U is monitoring V, it first verifies in the advertisement that V is not also monitoring U. If U and V are monitoring each other, a tie function is applied on their identifiers in order to elect the monitoring overlay node. If U stops monitoring V, it reacts as described above. If it continues to monitor V after the check, it sets $d(U, V)$ to the value h.

If $d(U, V)$ has changed, U corrects the position of V in L_U. If it is monitoring V, it also directly sends a new advertisement to V in order to inform it about the distance update (or, as explained in next section, a delete message indicating that the neighbour relation is no more useful).

Old unicast overlay neighbour leaving. Once the distance update process is completed, node U calculates the new position it would occupy in L_V consequently to the distance update. On the basis of the updated lists L_U and L_V, it can then detect if V is still required or not in L_U. If the position of V in L_U and of U in L_V are both greater than K, then the neighbourhood relationship between U and V is no more necessary. Node U deletes V from L_U and cancels its advertisement timer for V. It also sends an OTC_DELETE message to V. If the delete message is lost, it will be sent again at reception of the next advertisement from V.

If node V remains in L_U, node U can then determine if it must be selected as a neighbour or pruned.

3.5 The Selection of Neighbours in the List

A node U selects or prune a neighbour candidate V on the basis of the updated lists L_V and L_U. The pruning rule has a real parameter $\alpha \in [1, 2]$ that we call the *pruning selectivity*. An overlay link (U, V) is pruned if and only if there exists a third overlay node W that appears before V in L_U and before U in L_V, and such that $d(U,W)+d(W,V) \leq \alpha d(U,V)$. The pruned neighbours are not deleted from the candidates list. OTC control messages are continuously exchanged with pruned neighbour candidates as well as with selected ones. Oppositely, the overlay data messages will only flow on selected overlay links.

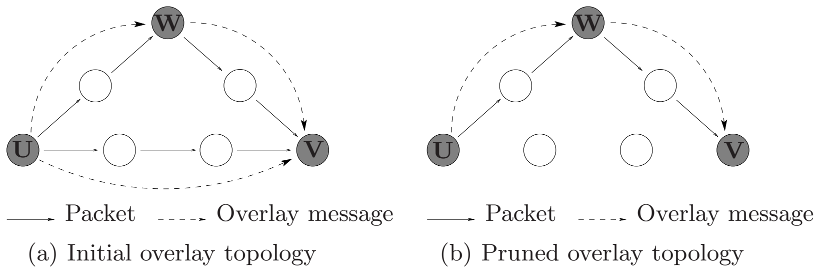


Fig. 2. Diffusion consumes less bandwidth on the pruned overlay topology

An example is given on Fig. 2. The underlay paths $U \rightarrow W$ and $V \rightarrow W$ are both shorter than $U \rightarrow V$. Nodes U and V thus prefer to select W than each other as neighbour. The idea is, for every pair of candidate overlay neighbours (U, V) to prune each other if they share a common better candidate, under the condition that the remaining overlay path that links them is not stretched by more than a factor α[2]. Notice that the pruning rule does not affect the overlay connectivity and that both overlay nodes U and V take locally the same pruning decision about each other. In this example topology, the number of packets emitted when U sends an overlay broadcast message is reduced from 7 packets on the initial topology to 4 on the pruned topology. However, the time needed for V to receive the first copy of the message and the path followed by this copy from U to V may be longer. We observed on larger topologies that the gain in bandwidth and its resulting decrease of contention were significant, while the mentioned increase of path stretch and diffusion time stayed acceptable.

[2] Note however that the final maximal overlay stretch may be higher than α because the pruning rule is also applied to (U, W) and (W, V), and so on.

4 Evaluation

4.1 Performance Criteria and Flooding Method

The objective of OTC is to offer a logical communication structure between the overlay nodes which allows the deployment of efficient overlay routing protocols and services. The quality of the overlay structure is thus strongly linked to desired properties of overlay routing protocols. We translate this in terms of the following objectives.

1. Diffusion bandwidth: As routing control traffic is often generated by flooding, the bandwidth necessary to send a message from one overlay nodes to all other ones, with a simple flooding procedure, must be as low as possible.
2. Diffusion time: In order to quickly compute valid routes, the overlay control traffic must be flooded rapidly.
3. Diffusion delivery percentage: In order to find routes, the overlay control traffic must be received by all overlay nodes.
4. Path stretch: The average cost of the shortest overlay path between any pair of overlay nodes must be as close as possible to the cost of the direct underlay path between them.

In order to spare bandwidth, an overlay node employs the following flooding technique:

1. For all overlay neighbours located only one hop away, it emits a single overlay message, which is actually broadcast in the underlay with a *Time To Live* (TTL) field set to one.
2. For every overlay neighbour located further away, an individual overlay message is created, which will be unicast to it by the underlay routing protocol.

4.2 Simulations Description

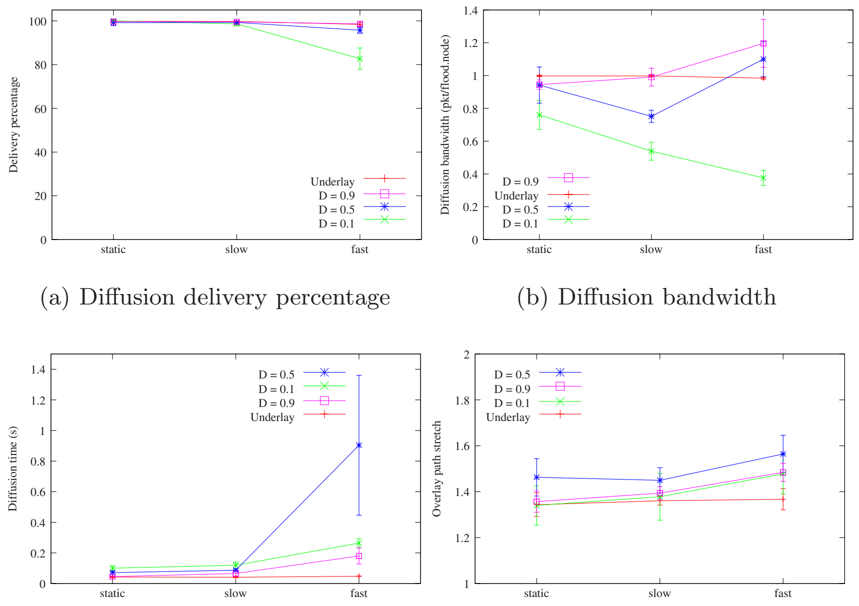
We use the ns-2.29 simulator. There are 100 ad hoc nodes, randomly and uniformly distributed on a square field. The length of the field is of 1200 meters and the radio transmission range of the nodes equals 250 meters. All experiments are conducted for overlay densities of 10, 50 and 90%. The simulation duration is of 150 seconds. The OTC protocol starts on every overlay node at the beginning of the simulation and uses a pruning selectivity $\alpha = 1.5$. In the static case, the maximal pruning selectivity provide the best results [14]. However, we observed with the mobile scenarios that some more resilience, afforded by a lower selectivity, is preferable when the nodes move in order to preserve the overlay structure connectivity. After 30 seconds, a source overlay node starts to emit 100 overlay messages of 64 bytes, at the average rate of one message per second, and the performance log also commences.

We test static and mobile scenarios. The latter are generated by the random waypoint scenarios generator provided with the ns distribution and grouped in two sets, slow and fast (as in [7]). Slow scenarios are defined by a pause time uniformly distributed in $[0, 10]$ seconds and a speed in $[1, 5]$ meters per second. The pause time of fast scenarios is uniformly distributed in $[0, 5]$ seconds and their speed in $[5, 15]$ meters per second.

We present the performance of OTC over the Optimised Link State Routing protocol (OLSR) [15], a proactive routing protocol. We also ran simulations over the Ad-hoc On-demand Distance Vector routing protocol (AODV) [16]. Results proved that the OTC protocol can be applied over reactive as well as over proactive routing protocols. However, we observed that the performance obtained over AODV rapidly degrades when node move, except for the lowest overlay density. The reason is that AODV cannot sustain the maintenance of many paths in a mobile context. Proactive routing protocols are less efficient than AODV when the traffic pattern is light, because the paths between each pair of ad hoc nodes is maintained even if only of few of them are required by the users. However, their routing load is far less sensitive to the number of user flows. When these are numerous, proactive routing protocols outperform reactive ones.

4.3 Results

The average performance obtained when flooding an overlay message on the topologies built by OTC over OLSR are shown on Fig. 3. The 95%-confidence intervals are specified. As a reference, we also indicate, on each sub-figure, the performance obtained when flooding a broadcast packet on the underlay.



(a) Diffusion delivery percentage

(b) Diffusion bandwidth

(c) Diffusion time

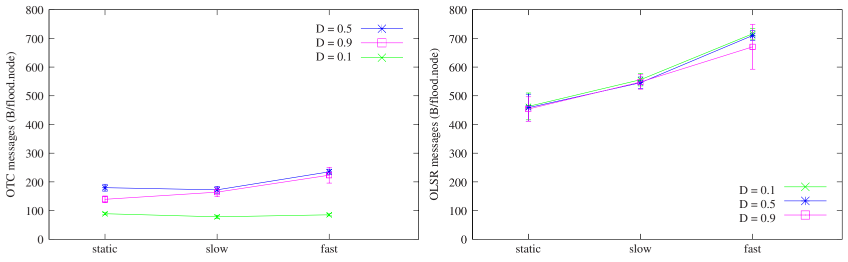
(d) Path stretch

Fig. 3. Overlay flooding performance over OLSR

Diffusion delivery percentage. For all overlay nodes, except the overlay broadcast messages source, we determined how many overlay messages out of the hundred sent were received before the end of the simulation.

The diffusion delivery percentage over OLSR is very good in the static and slow case, for all overlay densities. When nodes move a lot and rapidly, the diffusion delivery percentage is also very good (above 94%), except for the lowest overlay density (only 83% of the overlay nodes receive the flooded message). Noticeably, in the latter case, we obtained very good performance over AODV because the number of overlay links is low. In such situation, AODV resists better to mobility than OLSR [17]. Hence, we assume that the performance drop is due to OLSR and not to OTC.

Bandwidth consumption. We logged the number of packets emitted during the simulation. We classified them into three categories: the packets containing the 64 bytes of data, the OLSR control messages and the OTC control messages. The control bandwidth includes both OLSR and OTC messages. The metric used on Fig. 3(b) is the number of packets emitted per flood and per ad hoc node.



(a) OTC (b) OLSR

Fig. 4. Comparison of the bandwidth consumed by OTC and OLSR

The total bandwidth amount increases reasonably with the degree of mobility. At very low and intermediate overlay densities, the flooding of a message on the overlay, which is able to propagate useful information for overlay routing, costs less than one packet per node. For all cases studied, we observed that the OTC traffic occupies less than the third of the total amount of control traffic. Figure 4 shows that maintaining an overlay with OTC over OLSR costs less than half the bandwidth needed by OLSR for maintaining underlay routes. The metric used is the number of bytes emitted per flood and per ad hoc node.

Diffusion time. For each overlay message flooded on the overlay, we logged the interval of time elapsed between its emission and the moment at which its first copy was received by the last overlay node. This measures the maximum time that would be needed for an overlay route request to reach any destination.

The overlay diffusion time raises with the mobility degree. The sharper increase is observed at the middle overlay density because of our flooding policy, where each overlay node must emit a new message for each of its unicast neighbours. At a high overlay density, most neighbours are located at one hop (they are broadcast neighbours). At low density, there are a few overlay links, thus the time required for accessing the media is lower.

Path stretch. Each time an overlay message was received, we also computed the ratio of the number of hops it has passed through since its emission and of the shortest path length from the source. This defines the overlay path stretch. Its average value is under 1.6 for all cases studied. Note that the path stretch of broadcast packets flooded on the whole ad hoc network is also greater than 1.0 because of collisions.

5 Conclusions

We described and evaluated a protocol for building and maintaining a generic service overlay over an ad hoc network. The OTC protocol keeps, in a mobile environment, a set of overlay links as close as possible to a target overlay topology defined in previous work [14].

While service overlays have proved their utility in the Internet, there are currently only a few proposals for ad hoc overlays. We used a general approach. The overlay maintenance protocol is not application-specific, avoids as much as possible cross-layer optimisations and assumptions about the underlay topology or routing protocol. The performance evaluation utilises generic performance criteria based on overlay flooding.

On top of OLSR, the maintenance of the overlay generates an acceptable volume of OTC messages. The flooding of a message on overlay topologies built by OTC always shows up good performance, except when the overlay density is very low and the mobility degree high.

Note that the OTC protocol is not proposed as a final, ready-to-use solution. The performance study mainly shows the feasibility of maintaining the overlay structure while using a minimal amount of information. Our proposition thus leaves a large open space for optimisations.

Acknowledgements

This work has been partially supported by the European Union under the ANA FET project (FP6-IST-27489).

References

1. Bavier, A., Feamster, N., Huang, M., Peterson, L., Rexford, J.: In vini veritas: realistic and controlled network experimentation. In: Proc. of ACM SIGCOMM (2006)
2. Li, Z., Mohapatra, P.: Qron: Qos-aware routing in overlay networks. IEEE Journal on Selected Areas in Communications 22(1) (2004)

3. Subramanian, L., Stoica, I., Balakrishnan, H., Katz, R.H.: Overqos: offering internet qos using overlays. SIGCOMM Comput. Commun. Rev. 33(1) (2003)
4. Touch, J.: Dynamic internet overlay deployment and management using the x-bone. Computer Networks 36(2-3) (2001)
5. Klein, M., Konig-Ries, B., Obreiter, P.: A lightweight overlay for service discovery in mobile ad hoc networks. In: Proc. of 3rd Workshop on Applications and Services in Wireless Networks (ASWN 2003) (July 2003)
6. Xie, J., Talpade, R.R., McAuley, A., Liu, M.: Amroute: Ad hoc multicast routing protocol. Mobile Networks and Applications 7(6) (2002)
7. Conti, M., Gregori, E., Turi, G.: A cross-layer optimization of gnutella for mobile ad hoc networks. In: Proc. of ACM MobiHoc 2005, Urbana-Champaign, IL (2005)
8. Delmastro, F.: From pastry to crossroad: Cross-layer ring overlay for ad hoc networks. In: Proc. of 2nd IEEE International Workshop on Mobile Peer-to-Peer Computing (MP2P 2005) (March 2005)
9. Pucha, H., Das, S.M., Hu, Y.C.: How to implement dhts in mobile ad hoc networks? In: Proc. of MobiCom 2004, Philadelphia, PA (2004)
10. Cramer, C., Fuhrmann, T.: Proximity neighbor selection for a dht in wireless multi-hop networks. In: Proc. of 5th IEEE International Conference on Peer-to-Peer Computing (P2P 2005) (August 2005)
11. Zahn, T., Schiller, J.H.: Madpastry: A dht substrate for practicably sized manets. In: Proc. of 5th Workshop on Applications and Services in Wireless Networks (ASWN 2005) (June 2005)
12. Pucha, H., Das, S.M., Hu, Y.C.: Ekta: An efficient dht substrate for distributed applications in mobile ad hoc networks. In: Proc. of 6th IEEE Workshop on Mobile Computing Systems and Applications (WMCSA 2004) (December 2004)
13. Kummer, R., Kropf, P., Felber, P.: Distributed lookup in structured peer-to-peer ad-hoc networks. In: Proc. of International Conference on Distributed Objects and Applications (DOA 2006) (October 2006)
14. Calomme, S., Leduc, G.: Efficient and resilient overlay topologies over ad hoc networks. In: Hutchison, D., Katz, R.H. (eds.) IWSOS 2007. LNCS, vol. 4725, Springer, Heidelberg (2007)
15. Clausen, T., Jacquet, P.: Optimized link state routing protocol (olsr).
16. Perkins, C., Royer, E.M.: Ad hoc on-demand distance vector routing. In: Proc. of IEEE Workshop on Mobile Computing Systems and Applications(WMCSA 1999) (1999)
17. Clausen, T., Jacquet, P., Viennot, L.: Comparative study of routing protocols for mobile ad-hoc networks. In: Proc. of 1st Annual Mediterranean Ad Hoc Networking Workshop (Medhocnet 2002) (2002)