Topology Management for Improving Routing and Network Performances in Mobile Ad Hoc Networks

Navid Nikaein and Christian Bonnet
Institut Eurécom
2229, Route des Crêtes, B.P. 193
06904, Sophia Antipolis, France
Email: Navid.Nikaein, Christian.Bonnet@eurecom.fr
Phone: +33(0)4.93.00.26.74 & +33(0)4.93.00.26.08
Fax: +33(0)4.93.00.26.27

***Abstract*— A distributed topology management algorithm based on the construction of a forest from the topology of the network is proposed. In this algorithm, each tree of the forest forms a zone, and each zone is maintained proactively. As a result, the network can be seen as a set of non-overlapping zones. We introduce the concept of quality of connectivity for extracting the links connecting the pair of best nodes, and use this quality to construct the forest. We characterize the behaviors of the proposed topology management algorithm under various network density. We study the effect of the topology management on the performance of a ad hoc routing protocol. The results demonstrate that the performance of routing can be significantly improved with the help of topology management.**

***Index Terms*— Mobile ad hoc networks, architecture, topology management, network topology, simulation, performance evaluation.**

I. INTRODUCTION

A mobile ad hoc network (Manet) consists of a collection of mobile nodes forming a dynamic autonomous network through a *fully mobile infrastructure* [1]. Nodes communicate with each other without the intervention of centralized access points or base stations. In such a network, each node acts as a host, and may act as a router. Due to the limited transmission range of wireless network interfaces, multiple *hops* may be needed to exchange data between nodes in the network, which is why the literature sometimes uses the term *multihop* network for a Manet. The *topology* of a multihop network is the set of communication links between nodes used by a routing mechanism. Weeding out redundant and unnecessary topology information is usually called *topology management*. The topology management plays a key role in the performance of a routing protocol simply because the wrong topology information can considerably reduce the capacity, increase the end-to-end-delay and routing control overhead, and decrease the robustness to node failure.

There are two approaches to topology management in mobile ad hoc networks: power control and clustering [2]. Power control mechanisms adjust the power on a per-node basis, so that one-hop neighbor connectivity is balanced and overall network connectivity is ensured [3], [4]. R. Ramanathan et al. proposed to adjust incrementally node transmit powers in response to topological changes so as to maintain a connected topology using minimum power [5]. However, topology derived from power control schemes often result in unidirectional links that create harmful interface due to the different transmission ranges among one-hop neighbors [6]. In this paper, we focus on clustering approach, with which a subset of the set of nodes, called clusterhead, is selected to serve as the network backbone [7], [2]. Every node is then associated with a clusterhead, and clusterheads are connected with each other via gateway nodes. Therefore, the union of clusterheads and gateway nodes constitute a connected backbone. For clustering to be effective, the links and nodes that are part of the backbone must be close to minimum and must also be connected. In graph theory, the minimum dominating set (MDS) problem and the relevant minimum connected dominating set (MCDS) problem best describe the clustering approach to topology management. The problem of computing the minimum dominating set is known to be NP-hard even when the complete network topology is available [8]. In the heuristics that have been proposed, the selected clusterheads are equivalent to MDS, and the union of gateway nodes and clusterheads forms a connected dominating set (CDS), which is sub-optimum solution of the MCDS problem. Clusterheads can be elected via non-deterministic negotiations or by applying deterministic criteria [2]. Negotiations require multiple incremental steps, and may incur an election delay due to the lack of consensus about the nodes being elected as the clusterheads. Core extraction algorithm is one of the example of this approach [9]. In contrast, deterministic criteria can determine the clusterhead in a single round. Several approaches apply the node identifiers (ID) [8], [10], the node degrees [9], [11], or a combination of node's status (e.g. battery life and mobility rat) [2] as the criteria to elect the clusterhead within one or multiple hops. The choice of the criteria is very critical because it determines how the algorithm behaves to the different conditions that may be encountered in Manet (e.g. mobility rate and traffic load). For example, the node ID-based algorithms perform better that the degree-based algorithms in terms of cluster stability [12], but it fails to achieve load balancing in the network [13].

In this paper, we suggest a distributed clustering algorithm with deterministic criteria for topology management in mobile ad hoc networks. The main idea of the algorithm is to select for each node a neighbor, called *preferred neighbor*, that has a maximum degree of connectivity in the neighborhood (i.e. criteria of election algorithm). This is done using only periodical beaconing process. It has been proven that whatever the network topology is, connecting each node to its preferred neighbor always yield to a forest [14]. In this algorithm, each tree of the forest forms a zone, and each zone is maintained proactively. Therefore, the network is partitioned into a set of non-overlapping zones. As a result, the algorithm combines two notions: forest and zone. Forest reduces the broadcasting overhead by selecting a subset of the set of neighboring nodes for forwarding a packet, and zones are used to reduce the delay due to routing process and to reach high scalability. We also propose a mechanism to describe the quality of connectivity (QoC) for extracting the links connecting the pair of best nodes over time from the network point of view, and use this as the criteria for the election algorithm [15]. This is because the performance of ad hoc routing strictly depends on the quality of each individual node. Indeed, this quality should not only reflect the available resources at a node but also the stability of such resources because of: first, mobile ad hoc networks potentially have less resources than wired networks and second, mobility may result in link failure. Therefore, by changing the criteria of preferred neighbor election from degree of connectivity to quality of connectivity, we construct a forest of nodes with the high quality links. This is desirable because the subset of the set of forwarding nodes belongs to the nodes with the high quality, which in turn improves routing performance.

The rest of the paper is organized as follows. Section II describes the clustering algorithm used for topology management. In section III, we propose a new criteria for election algorithm that takes into account node mobility and traffic load. In section IV, we characterize the behavior of the algorithm under various network densities. Section V highlights the application of topology management to ad hoc routing. Section VI presents protocol model and the simulation model used to evaluate the performance of the algorithm under various network loads and mobility rates, followed by the results obtained by the simulation. Finally in the last section, we draw concluding remarks and highlights some future work.

II. Topology Management in Mobile Ad Hoc Network through a Clustering Algorithm

The algorithm consists of seven cyclic time-ordered phases: (A) neighboring table construction, (B) preferred neighbor election, (C) forest construction, (D) intra-zone clustering, (E) inter-zone clustering, and (F) zone maintenance which are executed based on the information provided by beacons. A beacon is a periodic message exchanged *only* between a node and its neighboring nodes. The content of a beacon is primitive at the beginning, and it will be enriched during subsequent phases of the algorithm. During the beaconing process, each node gathers the information describing its neighborhood in its neighboring table. Then, each node in the network topology chooses a neighbor whose degree of connectivity is equal to the maximum neighborhood degree. This neighbor is called preferred neighbor, and it is used to construct a preferred link between nodes for the purpose of routing. Then, a forest is constructed by connecting each node to its preferred neighbor and vice versa. Afterward, the intra-zone clustering algorithm develops a set of non-overlapping zone each of which is a tree, and builds the intra-zone routing table. The inter-zone clustering algorithm provides the inter-zones connections which are kept in the inter-zone routing table. Since the forest is abstracted into a set of non-overlapping zones, hence the network is partitioned. Finally, the zone maintenance phase updates the forest and the zones according to the preferred neighbor election phase algorithm. It has to be mentioned that the algorithm only uses beacons to perform every seven phases of the algorithm. Therefore, it avoids global broadcasting throughout the network.

A. *Neighboring Table Construction*

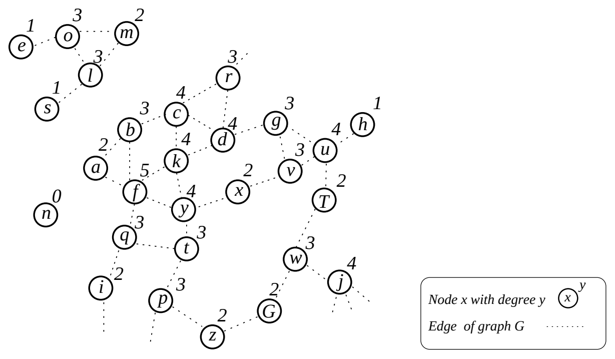
Basically neighboring table is the table through which a node detects changes to its neighborhood. This table consists of two information: a neighboring ID (NID) and its degree of connectivity (Deg) representing the number of neighbors. In order to construct this table, each node periodically broadcasts a beacon indicating its presence and its degree. Upon receiving a beacon, a node can gather information describing its neighborhood. Such information is considered valid for a limited period of time, and must be refreshed periodically to remain valid. Expired information is purged from the table. For example in Fig. 1, the neighboring table of node k has four entries, which are $< (f,5),(c,4),(d,4),(y,4) >$.

B. *Preferred Neighbor Election*

Let x and y be any nodes of the graph $G = (V,E)$. Based on the information provided by the neighboring table i.e. NID & Deg, node x can determine its preferred neighbor (PN). Fig. 1(a) shows an arbitrary graph, where each circle represents a node with its ID number and its degree (i.e. number of neighbors), and each dotted line represents an edge (or a link) between two nodes in the transmission range of each other (see legend in Fig. 1(b)). In order to determine the preferred neighbor, node x computes a set of nodes whose degree of connectivity are equal to maximum neighborhood degree. This set is denoted by

$PN_x = \{y | y \in N_x \wedge deg(y) = \max(deg(N_x))\}$, where N_x is the neighboring nodes of node x. We distinguish three cases:

- ***No PN***— if the set is empty, then node x has no PN which means it has no neighbors. In Fig. 1, node n has no neighbor and consequently no PN;
- ***Single PN***— if PN_x has only one member, then this member is the elected PN. For example, in Fig. 1, node k has four neighbors: f, c, d, y, but the set PN_k only includes f;
- ***Multiple PN***— the set PN_x can have more than one member which is the case for node f, since $PN_f = \{k, y\}$. This means that there are more than one neighbor with the maximum neighborhood degree. In this case, we assume that node x elects a node with the greatest ID number. So, node f elects node y since its ID number is greater than node k (regarding to the alphabetical order).



(a) An arbitrary graph G (b) Legend

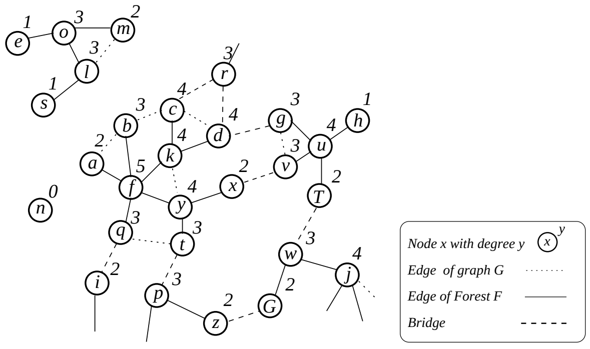
Fig. 1. Example

Consequently the main idea of the algorithm is to select, for each node x, a neighbor that has the maximum degree of connectivity in the neighborhood. For nodes that evaluate two identical values of degree, we break ties by setting the convention that nodes with higher IDs are preferred. We say that node y is the *preferred neighbor* of node x, if y is in the neighborhood of x and has the maximum degree among its neighbors. Therefore, each node elects exactly one PN and can be chosen as the PN of many nodes. Section III introduces alternative criteria for preferred neighbor election algorithm, which incorporate link quality.

C. Forest Construction

The forest is constructed by connecting each node to its PN. This is because of the way in which a node is elected follows a *monotonic increasing function* depending on the degree and on the ID number. It has been proven that whatever is the network topology, this approach always yields a forest (i.e. no cycle) [14]. Fig. 2(a) shows the constructed forest. In this figure, the solid lines represent the edges of the forest (or tree), and the dashed lines represent the edges connecting nodes that belong to different trees; which we refer to as gateway nodes (see legend Fig. 2(b)). These nodes are called gateway nodes, and will be used during inter-zone clustering phase (c.f. section II-E). A link is called *preferred link* if it connects a node to its preferred neighbor. The set of preferred links forms a set of preferred path in the network, which will be used during the routing process. Each node generates a table called *intra-zone table* (see Table II(a) & II(b)), which is periodically updated upon receiving beacons. This version of the table represents only the node IDs (NID) of the preferred neighbors that are available up to this phase. However, it will be extended, by adding another column called Learned_PN, when further information is available (see Table III(a) & III(b)).

The mechanism to build a preferred link between node x and y is as follows. When node x determines its PN y, it must notify its neighboring nodes, especially y, of its decision. Therefore, node x sets its beacon to $B_x = (x, deg(x), < 1 >, y)$, and updates its intra-zone table regarding y. This beacon indicates that node x with the degree $deg(x)$ is electing $< 1 >$ node y as its PN. This type of beacon is called beacon in election mode and is transmitted *periodically*. Upon receiving x's beacon, node y checks whether it has been chosen as the PN of x. If so, it also updates its intra-zone table regarding x. This means that a preferred link is built between node x and its preferred neighbor y, and thus node x and y belong to the same tree.



(a) The constructed forest (b) Legend

Fig. 2. Example

For example in Fig. 2, node k elects node f as its PN because of the highest degree. Then it generates the beacon $B_k = (k, 4, 1, f)$, and inserts node f into the NID field of its intra-zone table (c.f. Table II(a)). Upon receiving k's beacon, node f also inserts node k into its intra-zone table $Intra\_ZT_f$. Therefore, the link between node k and f becomes a preferred link. Node c and d will also elect node k as their PN, which is why node k inserts them into its intra-zone table. Similarly node y, b, q, and a will elect node f as their PN. In has to be mentioned that node f chooses node y as its PN, and it has been chosen as the PN of k, b, a, y and q. Table II(a) and II(b) show the intra-zone tables of node k and f; respectively.

TABLE I

INTRA-ZONE TABLE OF NODES k AND f REGARDING FIG. 2

NID
f
c, d

(a) $Intra\_ZT_k$

NID
y
k
b, a, q

(b) $Intra\_ZT_f$

D. Intra-zone Clustering

At this phase, nodes attempt to expand their own view about the tree they belong to by completing their intra-zone table. Indeed, each node locally advertises the new PN learned during the forest construction phase in terms of a new type of beacon called beacon in *forward* mode. Upon receiving this beacon, each tree member determines the *new* PNs learned from that neighbor and re-advertises to their neighbors if it is not a leaf node. For this purpose, each node generates another field in their intra-zone table called *Learned_PN* in order to keep the nodes that will be learned to be a tree member by their corresponding NID. We remind that, node y is chosen to be the PN of x, and x has sent a beacon to inform its neighborhood of its elected PN. Among the neighboring nodes of x, the PN y forwards x's decision to nodes that hold a tree edge with y[1] by setting its beacon to $B_y = (y, deg(y), < 0 >, x)$. If node y is chosen as the PN of many nodes through a period, then y forwards their decisions encapsulated in PN field in the beacon, that is $B_y = (y, deg(y), < 0 >, x : x' : x'' : ...)$. This beacon indicates that mode y with the degree $deg(y)$ is forwarding $< 0 >$ the new learned PNs or tree members $x : x' : x'' : ..$ to the tree, and it is called beacon in forward mode. It has to mentioned that this beacon is a *non-periodical* beacon. Other neighboring nodes of x add y to the Learned_PN field corresponding to x if they belong to the same tree as node x (i.e. node x belongs to their intra-zone tables). In this way, we say that y is learned to be the PN of x. Note that node x is also learned by the neighboring nodes of y. Node x generates a beacon in the PN forward mode if the set of PN learned by x is non-empty. This set is denoted by $Learned\_PN_x$. Node x transmits the beacon in forward mode with the $Learned\_PN_x$, if it is not a leaf node. Hence, the intra-zone table is *only* updated through the *immediate* preferred neighbor.

For example in Fig. 3, node k elects node f as its PN. So, node f can be learned by nodes c, d. Upon receiving k's beacon, node f updates the information regarding node k in $Intra\_ZT_f$. Since nodes b, a, q, y, k choose node f as their PN, node f must forward their decisions encapsulated in PN-field by setting its beacon to $B_f = (f, 5, 0, b : a : q : y : k)$. Therefore, nodes b, a, q, y are learned by node k, i.e. $Learned\_PN_k = b, a, q, y$, and are inserted into the Learned_PN field corresponding to f (see Table III(a)). Similarly, the set of $Learned\_PN_k$ will be learned by the nodes c & d. However, c, d will not forward their learned PN, since they are leaf nodes. Fig. 3 shows how the network is partitioned into a set of non-overlapping zones. Table III(a) and III(b) represent the intra-zone tables of nodes f and k, when their trees are established.

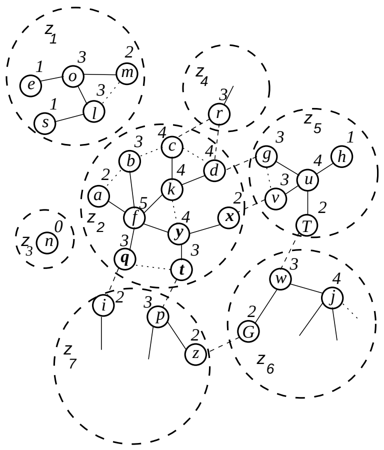


Fig. 3. Zone abstraction

TABLE II

INTRA-ZONE TABLE OF NODES k AND f REGARDING FIG. 3

NID	Learned_PN
f	a, b, q, y, t, x
c, d	-

(a) $Intra\_ZT_k$

NID	Learned_PN
y	x, t
k	c, d
b, a, q	-

(b) $Intra\_ZT_f$

Consequently, the view of node x about its tree consists of two levels: NID, $Learned\_PN$. The NID level contains the nodes holding tree-edges with node x, i.e. node x can reach them directly. The second level $Learned\_PN$ contains the nodes that are learned by the NID level. In fact, node x can reach them via their corresponding NID in the intra-zone table. Therefore, node x only knows the next hop for the learned PN. Therefore, each entry in $Intra\_ZT_x$ can be viewed as a branch of x. Thus, each node obtains a partial view of its tree in the sense that it does not know the detailed structure of its tree. For example in Fig. 3, consider the scenario where node k wants to communicate to one of the nodes belonging to its tree. According to its intra-zone table (see Table III(a)), node k can reach the nodes a, b, q, y, t, x through node f, while other nodes f, c, d are directly reachable. So,

[1]These nodes reside in the first column of intra-zone table of node y, i.e. $Intra\_ZT_y.NID$.

regarding to $Intra\_ZT_k$, the next hop to reach the nodes a, b, q, y, t, x is node f and not c, d. Fig. 4 shows the view of node k on its tree.

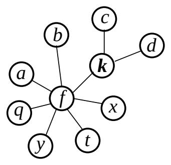


Fig. 4. View of node k about its tree

E. Inter-zone Clustering

Each node x encounters two cases during the construction of its tree. Firstly, it can succeed to add some nodes to its tree and updates its intra-zone table. Otherwise, node x puts the remaining nodes in its *inter-zone table*. These nodes are considered as gateway nodes and they will be moved from the inter-zone table to intra-zone table whenever they can join x's tree. For example in Fig. 3, node c belongs to the inter-zone table of node d until node d is informed about the existence of c in the tree. After that, node d moves node c from its inter-zone table to its intra-zone table.

F. Zone Maintenance

The topology information provided by the algorithm is strictly related to the preferred neighbor election algorithm. Such information may change as the preferred neighbor changes. There exist two cases where such information changes: (i) the current PNs expire, and (ii) a new PN is determined. To handle the former case, node x periodically determines the expired PNs. A PN expires if no beacon is received from that PN within a beaconing period. In this case, when a PN expires, say y, the entire entry expires with the PN, i.e. $y \rightarrow Learned\_PN$, and hence they are removed from the table. If node x succeeds to remove one or several PNs and yet it is not a leaf node, then it generates a *remove* beacon to notify the remained tree members about the removed members. A remove beacon encapsulates the removed entries in the PN field of a beacon. For this purpose, node x sets its beacon to $B_x = (x, deg(x), < -1 >, y \rightarrow Learned\_PN)$, where $< -1 >$ means that each tree member has to remove the PNs. Upon receiving x's beacon, each neighbor updates its intra-zone table accordingly; and generate a new remove beacon if it is not a leaf node. In the latter case, once node x determines its *new* PN y', it verifies as to whether it has been chosen as the PN of y. If so, i.e. $PN_y == x$, no remove beacon is sent and the PN y remains in the intra zone table of node x since node x is the preferred neighbor of node y. Then, node x sends a PN election beacon as it is explained in section II-C, and the new PN of node x will be learned by the zone and specially by node y. Otherwise, if node x has not been chosen as the PN of y, then x removes the entire branch corresponding to its old PN y, i.e. $y \rightarrow Learned\_PN$, and notify the remained members of its tree about the removed members as in the former case. Afterward, node x sends a PN election beacon for its new PN. Note that each node periodically performs the preferred neighbor election.

For example in Fig. 3, if the link between nodes k and f is broken, then node k removes node f and its learned PNs $f \rightarrow a : b : q : y : t : x$ from its intra-zone table (see Table III(a)). Then, node k forwards the beacon $B_k = (k, deg(k), < -1 >, f \rightarrow a : b : q : y : t : x)$ to the remaining tree members which are node c and d. Nodes c and d do not forward the beacon, since they are leaf nodes. Also, node f cuts the branch corresponding to $k \rightarrow c : d$, and notifies the b, a, q, y about cut branch. Other tree members carry out the same procedure.

III. Quality of Connectivity

Due to frequent changes in the network topology and limited network resources, routing in Manet experiences link failure more often. As the degree of connectivity does not characterize the cause of link failure, we suggest a mechanism to describe the quality of connectivity (QoC) for extracting the *links* connecting the pair of best nodes over time from the network point of view, and use this quality as *the* criteria for preferred neighbor election [15]. Link failure stems from node mobility and lack of network resources. Therefore it is essential to capture the aforesaid characteristics to identify the QoC. We identify two metrics to represent the quality of connectivity from the network point of view: buffer level, and stability level. The quality of connectivity of a particular node reveals whether the node is *forced* to be in *non-router* mode. In the *non-router* mode, a node ceases to be a router and acts only as a host. In this paper for the sake of simplicity, we only consider a *homogeneous* environment where all nodes have similar capabilities such as transmission range and buffer capacity.

- ***Buffer Level***— which stands for the available unallocated buffer. Note that if the buffer level of a particular node is low, then this implies that a large number of packets are queued up for forwarding, which in turn implies that a packet routed through this node would have to experience high queuing delays. A high buffer level indicates that the corresponding node has no packets queued up for forwarding, while a low buffer level shows that the available buffer is less than 25 percent of its size. In the latter case, a node is in a non-router mode. Since there is a slight delay between the broadcast of this metric and its use, instantaneous buffer-level may be misleading. Hence, a node should maintain the average buffer-level such as exponentially weighted moving average (EWMA).
- ***Stability Level***— we define the connectivity variance of a node with respect to its neighboring nodes over time as the stability of that node. This metric is used to avoid unstable nodes to relay packets. We estimate the stability of a node x as:

$$stab(x) = \frac{|N_{t_0} \cap N_{t_1}|}{|N_{t_0} \cup N_{t_1}|} \quad (1)$$

N_{t_1} and N_{t_0} represent the nodes in the neighborhood of x at times t_1 and t_0 respectively. Note that, $t_1 - t_0$ denotes the time period in which nodes exchange beacons. A node is unstable if a large number of its neighbors change. Further, if most (or all) of the neighbors remain the same at the two times t_1 and t_0, then we call this node stable. Note that $N_{t_1} \cap N_{t_0}$ (the numerator of $stab(x)$) denotes the set of nodes that have remained in the neighborhood of x between times t_0 and t_1. The denominator of $stab(x)$ is a normalization term. A node has high stability if none of its neighbors change ($N_{t_1} = N_{t_0}$) , in this case we have $stab(x) = 1$. A node is *unstable* (no stability) and therefore in a non-router mode, if all its neighbors change ($N_{t_1} \cap N_{t_0} = \phi$), in this case we have $stab(x) = 0$.

In order to facilitate the notion of QoC, we need to map the QoC onto a single weighted metric which can be compared and whose best can be chosen. Suppose s and b denote the stability and buffer levels of a particular node. One way to estimate the QoC of node x is:

$$QoC(x) = f(s,b) = \alpha \cdot s + \beta \cdot b \quad (2)$$

The weights α and β denote the relative importance of stability and buffer amongst themselves. Since we desire stability to be the most important followed by the buffer level, we propose $\alpha = 2$ and $\beta = 1$. Hence, given two nodes, we are always in a position to select the *better* one. For example in Fig. 1 suppose that $s, b \in \{0, 10\}$, if a node f has $s = 6.5$, $b = 4.3$ then its QoC is: 17.3. On the other hand a node k with $s = 7.2$, $b = 8.1$ has a QoC value of 22.5. Hence, in our scheme, node k is a "better" node than node f. In this way, the algorithm extracts the high quality links and and adjust the network topology accordingly so as to reduce the probability of link failure.

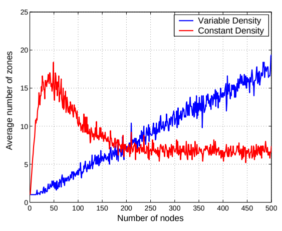
IV. Behavioral Results on the Topology Management Algorithm

We study the behavior of the ad hoc topology managemnt algorithm with parameters such as average number of zones in the network, average number of nodes in a zone (zone size), zone diameter (i.e. in terms of number of hops), and the ratio of path length obtained by the algorithm to the optimal length under various network density. The following results were obtained by implementing the statitic version of the algorithm in C++ and measuring the metrics after the population of mobile nodes was distributed uniformly on a grid of $2000m \times 2000m$ with each node having a transmission range of 250m. The key aspect of these measurements is that they depict how the algorithm behaves with an increasing number of nodes in the network. We consider two cases: *variable density* where the network is sparse at the beginning and becomes highly dense, and *constant density* where the area covered by the ad hoc network increases as the number of nodes increases.

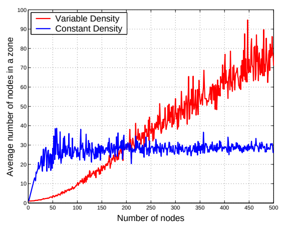
The graph in Fig. 5(a) shows the number of zones versus the number of nodes in the forest generated by the algorithm for a constant and a variable density zone partitioning. In a constant density zone partitioning method, the number of zones increases linearly as the number of nodes in the network increases. This means that the communication overhead within a zone remains constant as the number of nodes in the network increases at the expense of an extra overhead for the communication outside of the zone. However, the average number of nodes in a zone stays constant as it can be noticed in Fig. 5(b). In a variable density case, we observe that the more sparse the network is (below 200 nodes on a grid of 2000m $\times$ 2000m), the more partitioned the network becomes and hence the more number of zones the algorithm produces. As we increase the network density (from 200 up to 500) with the same size of the network, the number of zones remains constant and it tends to 7. This indicates that increasing network density has no effect on the number of zones but on the number of nodes in each zone as shown in Fig. 5(b). Therefore, in contrast to the former case, the communication overhead within zones linearly increases as the number of nodes increases but the communication overhead outside of zones remains constant.

The graph in Fig. 5(c) shows the diameter of a zone versus the number of nodes in the network. The diameter of a zone is defined as the length of the path that has the longest hop count. If the zone diameter is fixed, then we can place an upper bound on the end to end delay for a connection between two nodes belonging to the same zone. For this reason, it is preferable to maintain the zone diameter as low as possible even if the number of nodes increases. From Fig. 5(c) it can be inferred that even though the zone diameter increases initially as the number of nodes increases, it stabilizes for number of nodes greater than 200. The zone diameter determines the degree of trade-off between communication overhead and end-to-end delay of a routing protocol. The longer the zone diameter becomes, the more proactive the protocol is, and vice versa. As a result, when the zone diameter increases, the end-to-end delay decreases at the expense of an extra communication overhead.

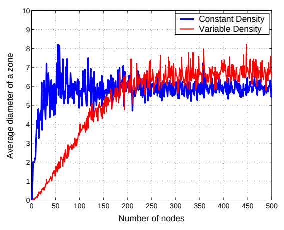
The graph in Fig. 5(d) measures the ratio of the tree hop count length to the optimal (shortest) hop count. This measurement is important since if the hop count of source, destination pairs in a forest is higher than that of the shortest hop path, then our protocol may suffer from consuming high network resources and incurring high delays. The ratio is measured using an average. That is to say that the measurement is averaged over all possible



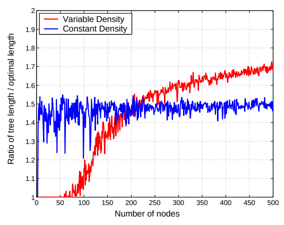
(a) The number of zones versus the number of nodes



(b) The size of zones versus the number of nodes



(c) The diameter of a zone versus the number of nodes



(d) The average ratio of the algorithm hop count to the optimal hop count vs the number of nodes

Fig. 5. Behavioral results of the topology management algorithm

source, destination nodes belonging to the same zone. Hence, if s and d are two mobile nodes belonging to the same zone in the tree T_i, and *Tree_Length* (s, d) denotes the number of hops on the path between s and d using the tree edges, and *Optimal_Length* (s, d) denotes the shortest number of hops, then the average measures:

$$\frac{\sum_{s,d \in T_i} \frac{\mathit{Tree\_Length}(s,d)}{\mathit{Optimal\_Length}(s,d)}}{\sum_{s,d \in T_i} 1} \tag{3}$$

As is observed from Fig. 5(d), we can deduce that on the average, the tree-length is no worse than twice the optimal shortest hop path for up to 400 mobile nodes in the topology. Furthermore, the trend of the graph suggests that this ratio is stable. This is a desirable result because the construction of the forest does not consider optimal hop count as a route determination metric to generate the forest.

V. Application of Topology Management to Ad Hoc Routing

One of the challenges in ad hoc routing is related to the underlying broadcasting technique or a derivative of it used to perform the routing functionalities. Most of the protocols both proactive and reactive employ the simplistic form of broadcasting called flooding, in which each node retransmits the unique received packet exactly once [16], [17]. This potentially generates high overhead in the network. One of the problems related to the flooding is the broadcast storm problem [18], where a node receives the same message from multiple neighboring nodes at about the same time. Observations in [18] reveal that serious redundancy, contention, and collision could exist if flooding is done blindly. Some methods are required to damp the process of packet generation and duplication at each node. One of the main application of topology management consists of carrying out reliable broadcasts in Manet, such that each message from a source node reaches every other node reliably. That is why we study the effect of the proposed topology management on ad hoc routing protocols, in particular on the hybrid protocols.

VI. Performance Evaluation

We use a simulation model based on *ns-2* in our performance evaluation, which is a discrete event simulator developed by the university of California at Berkeley and the VINT project [19]. The decision to use *ns* rather than the other simulators such as OPNET and GloMoSim (QualNet) was made by our deeper familiarity with *ns* and its wide usage.

A. Routing Protocol Model

As a routing protocol, we use HARP - hybrid ad hoc routing protocol [20] to study the impact of the proposed topology management. HARP combines a proactive behavior within a zone with a reactive behavior between zones. In fact, routing is performed on two levels: intra-zone and inter-zone, depending on whether the destination belongs to the same zone as the forwarding node. Intra-zone routing relies on the proactive mechanism of the topology management algorithm. It is important to note that intra-zone routing does not have any route acquisition delay thanks to the zone partitioning algorithm. Inter-zone routing, on the other hand, applies the path discovery procedure to find the shortest path to the destination's zone. In inter-zone routing, we try to conceal the zone details and to look upon them as nodes (i.e. zone level routing). The aim here is to establish a connection from the source node's zone to the destination node's zone. Hence in inter-zone routing, we are looking for a path of bridge edges that connect the zone of the source node to the zone of the destination node. We refer to a link as a *bridge* if it connects nodes belonging to different zones. During the process of path discovery, intermediate zones/nodes record the routing information in their routing table so as to establish the reverse and forward path.

B. Traffic and Mobility Models

Our protocol maintains a transmission buffer of 64 packets. It contains all data packets waiting to be transmitted including packets for which route discovery has been started, but no reply has arrived yet. In order to prevent indefinite buffering of packets, packets are dropped if they wait in the transmission buffer for more that 10 simulated seconds. The beaconing period is 10 simulated seconds. We use traffic and mobility models similar to those previously reported using the same simulator [21], [22], [23]. Traffic sources are CBR (constant bit rate). The packet size is 512 bytes and the packet rate is 4 *packets/second*. The mobility model uses the *random way point* model in a rectangular field of $1500m \times 300m$. This model was first used by Johnson and Maltz in the evaluation of DSR [24], and was later refined by the same research group. In this model, each node begins the simulation by remaining stationary for a pause time. It then selects a random destination in the $1500m \times 300m$ space and moves to that destination at a random speed uniformly chosen from $(0, V_{max}]$, where V_{max} is the maximum speed of the simulation. This model is expected to maintain this average speed as the simulation progresses. Recently, Yoon et al. have shown that the random way point model in its current form fails to reach a steady state in terms of instantaneous average node speed, but rather the speed continuously decreases as simulation progress [25]. Also, Camp et al. provide valuable simulation results that illustrate the importance of choosing a mobility model in the simulation of an ad hoc network protocol [26]. Simulations are run for 900 simulated seconds for 50 nodes. The selected pause times, which affects the relative speeds of the mobile, are $0, 30, 60, 150, 300, 600$, and 900 seconds. For each pause time, we randomly generate 10 different mobility scenarios. So, each data point in the performance results represents an average of 10 runs.

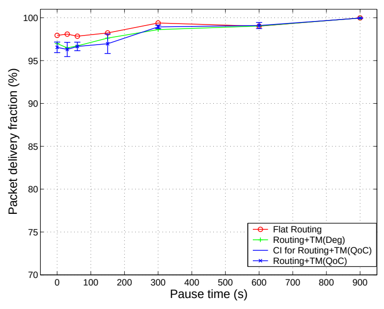
C. Performance Results

In the following, we compare the performance of the flat version of HARP (without any topology management strategy), with the ones that employ the topology management strategy. In the latter case, we vary the criteria for preferred neighbor election from degree of connectivity to quality of connectivity. In the following figures, these cases are distinguished by: *Flat Routing, Routing+TM(Deg)*, and *Routing+TM(QoC)* with its 95% confidence interval *CI for Routing+TM(QoC)* to differentiate each case. Three key performance metrics including *packet delivery fraction, average end-to end delay*, and *routing overhead* are evaluated under various traffic load and various mobility rate.

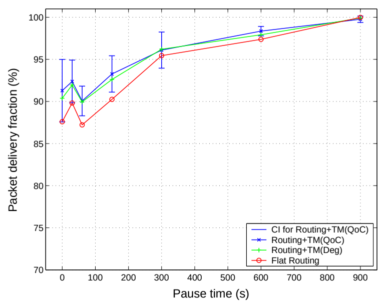
1) Packet Delivery Fraction: Packet delivery fraction is defined as the ratio of the data packets delivered to the destination to those generated by the CBR sources. Fig. 6 compares this metric for the *Flat Routing, Routing+TM(Deg)*, and *Routing+TM(QoC)*. In the low traffic load (10 sources), the *Flat Routing* has a slightly better packet delivery fraction than the two others (Fig. 6(a)). This is due to the shorter path length used in the *Flat Routing* and the fact that the network is not congested. Note that, in two other cases the average path length is no larger than 2 times of the optimal path length (c.f. Fig. 5(d)). However, the *Routing+TM* strategies outperform the *Flat Routing* by about 7 percent at lower pause time (high mobility) for the medium traffic load, and up to 20 percent as the pause time decreases for the high traffic load. This is because routing based on the minimum hop count metric always biases the same class of routes. As a results those routes become congested as the traffic load increases in the network. A similar phenomenon was also observed in [27]. However in the *Routing+TM*, this bias is reduced due to the dynamic nature of the forest and hybrid behavior of routing leading to the load balancing in the network (Fig. 6(b) & 6(c)). Furthermore, the *Routing+TM (QoC)* outperforms the *Routing+TM (Deg)* in the high traffic load. This is because of the high quality links used during the routing process. Note that the high quality links (or nodes) reduce the probability of link failure specially in the high mobility, and hence improve routing performance.

2) Average end-to-end delay of data packets: The average end-to-end delay includes all possible delays caused by buffering during route discovery latency, queuing at the interface queue, retransmission delays at the MAC, propagation and transfer times. As it is illustrated in Fig. 7, the *Routing+TM* approaches have *always* lower delay than *Flat Routing*. Indeed with medium and high traffic load, they improve the delay performance up to 50 percent for lower pause times. Again, this is due to the hybrid behavior of routing and the dynamic nature of the forest. Furthermore, *Routing+TM (QoC)* has even a better end-to-end delay since all forwarding nodes maintain high quality links. One interesting observation is that the delay increases with high traffic load with very low mobility rate. This is due to a high level of congestion and multiple access interfaces at certain regions of the network. Therefore, there is a need for a *load balancing* mechanism to distribute evenly the traffic when the network is congested. This phenomenon is less visible with higher mobility where traffic automatically gets more evenly distributed due to source movements. Note that both degree and quality of connectivity criteria for forest construction vary as a function of node mobility. Recall that in case of *TM (QoC)*, we biased the weight of stability by $\alpha = 2$ against the buffer by $\beta = 1$ (c.f. equation 2). A similar observation was also reported in [23], [22].

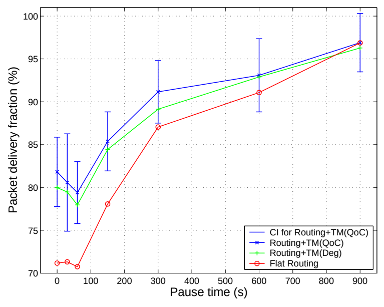
3) Routing overhead: Routing overhead is measured as total number of bytes and packets used for routing process during the simulation. The packet and byte overhead are shown in Fig. 8 & Fig. 9. Both *Routing+TM* protocols (Deg and QoC) have higher overhead than *Flat Routing* in low traffic load. This is due to the beaconing process used in these two approaches. Indeed, the process of topology management attempts to detect and react to link failures before its occurrence by adjusting the topology specially in case of the *Routing+TM (QoC)* protocol (see Fig. 9(d) & 8(d)). However in medium and high traffic load, the overhead for *Routing+TM* and *Flat Routing* are very similar specially in lower pause time. The reason is that the forest structure reduces the broadcasting overhead by



(a) 10 sources

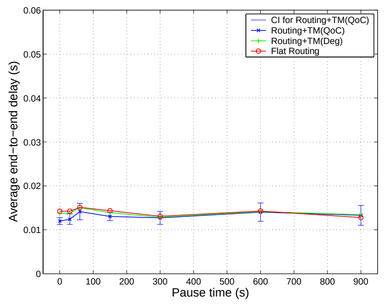


(b) 20 sources

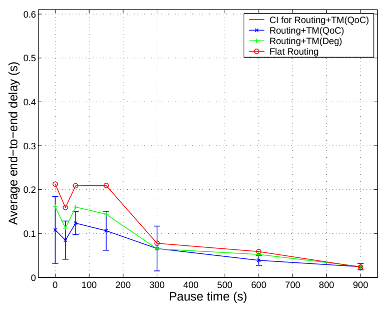


(c) 30 sources

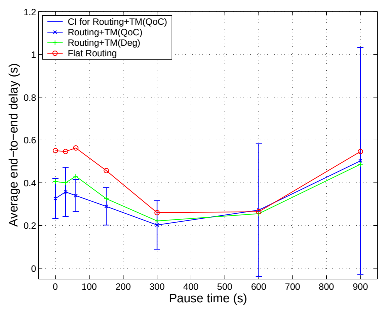
Fig. 6. Packet delivery fraction for the 50 nodes with various number of sources



(a) 10 sources



(b) 20 sources



(c) 30 sources

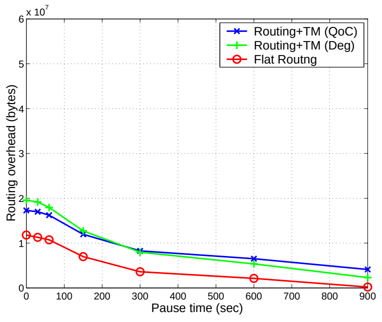
Fig. 7. Average data packet delay for the 50 nodes with various number of sources

selecting a subset of the neighboring nodes for forwarding a packet.

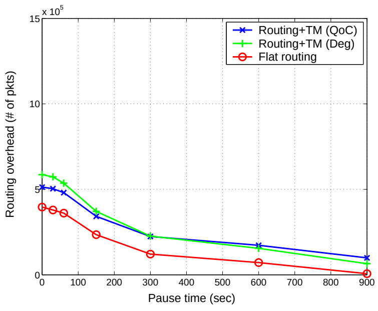
VII. Conclusion

We have proposed a novel topology management algorithm, which combines three main concepts: forest, zone, and quality of connectivity. Zones were used in order to reduce the delay due to routing process and to reach high scalability. Forest have reduced the broadcasting overhead by selecting a subset of neighboring nodes for forwarding a packet. The quality of connectivity is used to extract the links connecting the pair of best nodes to adapt the topology to the network conditions, i.e. traffic loads and mobility rates.

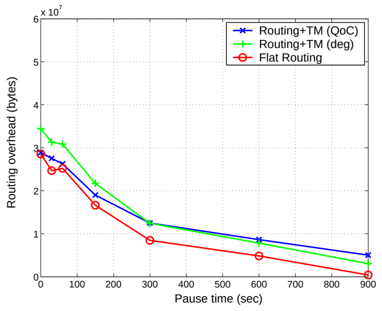
We have shown that the performance of routing is significantly improved with the help of topology management. The packet delivery fraction is improved up to 20 percent, and delay performance up to 50 percent. We have observed that routing protocols require a load balancing mechanism in order to evenly distribute the data traffic in the network. We have noticed the misbehavior of both criteria of preferred neighbor election as they fail to provide load balancing in the network when the rate of mobility is low. In future, we intend to evaluate the performance of the proposed topology management under various network size. Also, we aim to study the effect of different criteria for forest construction in order to achieve the load balancing in all network conditions.



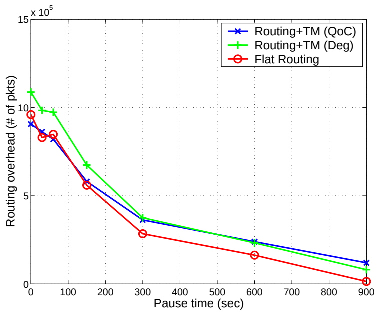
(a) 10 sources



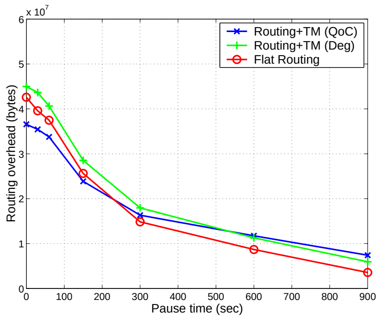
(a) 10 sources



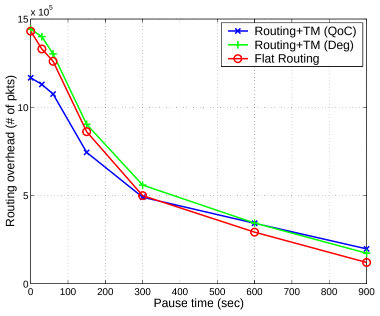
(b) 20 sources



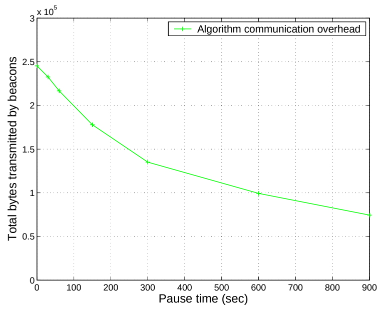
(b) 20 sources



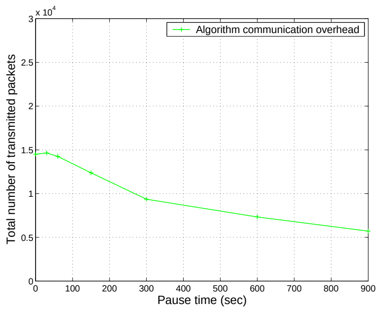
(c) 30 sources



(c) 30 sources



(d) Total transmitted bytes for TM (Deg)



(d) Number of transmitted beacon for TM (Deg)

Fig. 8. Routing overhead in terms of total transmitted bytes for 50 nodes with various number of sources

Fig. 9. Routing overhead in terms of number of packets for the 50 nodes with various number of sources

References

[1] *Mobile Ad Hoc Networking (MANET)*, Available at $http://tonnant.itd.nrl.navy.mil/manet/manet\_home.html$.

[2] L. Bao and J. J. Garcia-Luna-Aceves, "Topology management in ad hoc networks," in *Proceeding of MobiHoc*, 2003.

[3] H. Takagi and L. Kleinrock, "Optimal transmission ranges for randomly distributed packet radio terminals," *IEEE Transactions on Communications*, 1984.

[4] L. Hu, "Topology control for multihop packet radio networks," *IEEE Transactions on Communications*, 1993.

[5] R. Ramanathan and R. Rosales-Hain, "Topology control of multihop wireless networks using transmit power adjustment," in *Proceedings of INFOCOM*, 2000.

[6] "R. Prakash, "Unidirectional links prove costly in wireless adhoc networks," in *Proceedings of DIMACS Workshop on Mobile Networks and Computers*, 1999.

[7] P. Krishna, N. Vaidya, M. Chatterjee, and D. Pradhan, "A cluster-based appraoch for routing in dynamic networks," in *ACM SIGCOMM Computer Communication Review*, 1997.

[8] Alan D. Amis, Ravi Prakash, Dung Huynh, and Thai Vuong, "Max-min d-cluster formation in wireless ad hoc networks," in *Proceedings of INFOCOM*, 2000.

[9] R. Sivakumar, P. Sinha, and V. Bharghavan, "CEDAR: Core extraction distributed ad hoc routing," *IEEE Journal on Selected Areas in Communication*, 1999.

[10] M. Gerla and J. Tsai, "Multicluster, mobile, multimedia radio network," *Wireless Networks*, 1995.

[11] L. Jia, R. Rajaraman, and T. Suel, "An efficient distributed algorithm for constructing small dominating sets," in *Proceeding of ACM PODC*, 2001.

[12] C-C. Chiang, "Routing in clustered multihop, mobile wireless networks with fading channel," in *Proceedings of IEEE SICON*, 1997.

[13] A. Amis and R. Prakash, "Load-balancing clusters in wireless ad hoc networks," in *Proceedings of ASSET*, 2000.

[14] Navid Nikaein, H. Labiod, and C. Bonnet, "DDR-distributed dynamic routing algorithm for mobile ad hoc networks," in *Proceedings of MobiHOC*, 2000.

[15] N. Nikaein and C. Bonnet, "Improving routing and network performance in mobile ad hoc networks using quality of nodes," in *Proceedings of WiOpt*, 2003.

[16] B. Williams and T. Camp, "Comparison of broadcasting techniques for mobile ad hoc networks," in *Proceedings of MobiHoc*, 2002.

[17] J. Jetcheva, Y. Hu, D. Maltz, and D. Johnson, "A simple protocol for multicast and broadcast in mobile ad hoc networks," 1001, Internet Draft.

[18] Y.-C. Tseng, S.-Y. Ni, Y.-S. Chen, and J.-P. Sheu, "The broadcast storm problem in a mobile ad hoc network," in *Proceedings of ACM Wireless Networks*, 2002.

[19] K. Fall and K. Varadhan, NS *notes and documentation*, A Collaboration between researchers at UC Berkeley, LBL, USC/ISI, and Xerox PARC, available at: $http://www.isi.edu/nsnam/ns/$.

[20] Navid Nikaein, C. Bonnet, and Neda Nikaein, "HARP - hybrid ad hoc routing protocol," in *Proceedings of IST - International Symposium on Telecommunications*, 2001.

[21] J. Broch, D. A. Maltz, D. B. Johnsin, Y-C. Hu, and J. Jetcheva, "A performance comparison of multi-hop wireless ad hoc network routing protocols," in *Proceedings of MobiCom*, 1998.

[22] S. R. Das, C. E. Perkins, and E. M. Royer, "Performance comparison of two on-demand routing protocols for ad hoc networks," in *Proceedings of Infocom*, 2000.

[23] P. Johansson, T. Larsson, N. Hedman, and B. Mielczarek, "Routing protocols for mobile ad-hoc networks - a comparative performance analysis," in *Proceedings of ACM MobiCom*, 1999.

[24] D. B. Johnson and D. A. Maltz, *Dynamic Source Routing in Ad Hoc Wireless Network*, Kluwer Academic Publishers, 1996.

[25] J. Yoon, M. Liu, and B. D. Noble, "Random waypoint considered harmful," in *Proceedings of Infocom*, 2003.

[26] T. Camp, J. Boleng, , and V. Davies, "A survey of mobility models for ad hoc network research," in *Proceedings of WCMC, Wireless Communication & Mobile Computing*, 2002.

[27] D. S. J. De Couto, D. Aguayo, B. A. Chambers, and R. Morris, "Performance of multihop wireless networks: Shortest path is not enough," in *Proceedings of ACM SIGCOMM HotNets-I*, 2002.

Biographies

Navid Nikaein is a Ph.D. student in the Mobile Communication Department at Institut Eurécom, Sophia-Antipolis, France. He received the M.Sc. degree in Networking and Distributed Systems from Université de Nice Sophia-Antipolis (ESSI), France; and the B.Sc. degree in Mathematics Applied to Computer Science from Shahid Beheshti University (SBU), Tehran, Iran. His current research interests include routing, quality of service, and mobility management issues in mobile environments.

Christian Bonnet joined Institut Eurécom as an associate professor in 1992. Since 1998 he is at the head of the Mobile Communications Department of Eurécom. His teaching activities are distributed and real-time systems, mobile communication systems, wireless LANs and protocols for mobility management. His main areas of research are wireless protocols, wireless access to IP Networks and data communications in mobile networks including mobile ad hoc networks.