# Indoor experiments of self-organization and localization protocols for hybrid networks

Fabrice Theoleyre, Fabrice Valois

# Indoor experiments of self-organization and localization protocols for hybrid networks

Fabrice Theoleyre*† Fabrice Valois*,
* CITI - INSA Lyon / INRIA, 21 av Jean Capelle, 69621 Villeurbanne Cedex, France
† LIG - INPG / IMAG / CNRS, 681 rue de la passerelle, 38402 St Martin d'Heres Cedex, France

***Abstract*—MANET protocols are often evaluated through simulations, not in a real radio environment. We describe here the implementation and deployment of a complete testbed for hybrid networks, allowing a seamless integration of an ad-hoc network in the Internet. In particular, a self-organization protocol and the mobility management protocol benefiting from this organization were implemented. The performances demonstrate the feasibility and usefulness of this scheme. Besides, this testbed offers a detailed description of the requirements to constitute a wireless testbed and to test any protocol for ad hoc or hybrid networks.**

## I. Introduction

Mobile Ad Hoc Networks (MANET) are literally networks *ready to work*. All terminals can communicate with other nodes via wireless communications: MANET are spontaneous networks, without any fixed infrastructure. The network must function autonomously, without any human intervention: the self-configuration property is vital. In consequence, the nodes must collaborate to set up all network functions like routing. Because a source can be not in the radio range of the destination, intermediary nodes must relay the packets along a multihop path, each node being both router and client. Thus, a distributed routing algorithm must be proposed, computing efficient routes and dealing with network dynamicity. Ad-hoc networks connected to the Internet are often called hybrid networks, constituting multihop cellular networks. The most static nodes will surely forward the packets of the mobile nodes. In this paper, we focus our work on how to provide an efficient hybrid network, connected to the Internet, and test in one scenario the Internet access of a mobile node.

Recent works propose to self-organize the network via a virtual structure before its utilization. Such a self-organization is useful to give a macroscopic and stable view of the topology. Moreover, the introduced hierarchy allows to deploy more efficient protocols. Self-Organized Mobility Management protocol (SOMoM) [1] proposes a routing scheme to interconnect seamlessly a MANET to the Internet thanks to a self-organization. Routing caches are distributed in the backbone, to create stable routes and to reduce the overhead: a self-organization seems promising.

The protocols for ad hoc networks are mainly evaluated through simulations, like SOMoM was. Simulations present several assets in terms of reproducibility, financial cost, deployment ease and flexibility. However, simulations simplify the radio environment: fading, shadowing, reflections or diffractions cannot be well modeled. A tradeoff between accuracy and execution speed is therefore required.

The contribution of this article is to present a performances evaluation of both a self-organization and a routing protocol for hybrid networks. The performances are measured in a real indoor radio environment, reflecting the expected performances of an operational hybrid network. Thus, this solution proposes the creation of a complete multihop cellular network. In a more general manner, this article details the complete network architecture and proposes consequently the description of a generic testbed, reusable to evaluate the performances of any protocol for hybrid networks. This approach could constitute the first step to set up an efficient wireless testbed for any research in this domain.

This work is organized as follows. Section II presents a panorama of existing testbeds for wireless networks. Section II-A gives a short overview of the protocol implemented here, and section III details the design and the implementation of our testbed. Results are given in section IV and a discussion about the issues of current testbeds in section V. Finally, section VI concludes this article and gives some perspectives.

## II. Related Work

[2] presents a pioneering work in the conception of a testbed for MANET, measuring the performances of DSR, mainly the delays and TCP throughput. The authors propose also a MAC filtering to set up any topology. Similarly, [3] proposes to deploy cables, shields and signal attenuators to control the signal propagation: the reproducibility is total. However, such a testbed cannot model a real-world environment, limiting the utilization to debugging. In [4], AODV exhibits a 50% dropping rate in their outdoor environment, which is much lower than the performances obtained here. Recently, [5] presents a complete platform of 37 nodes. Approximatively 10% of the pairs of nodes do not find any route because of the lack of reliability of broadcasts. Recently, [6] proposed a survey of the current testbeds used in the scientific community.

### A. Overview of SOMoM

In [7], we introduced a self-organization based on a virtual backbone: some mobile nodes are elected to form a connected structure. The backbone consists in a tree structure, the Internet gateway being the root of this tree. Procedures allow to react to topology changes and to reconstruct locally the structure when it is broken. In particular, some control packets are periodically flooded in the backbone tree by the root to maintain the connectivity. In the future such packets will

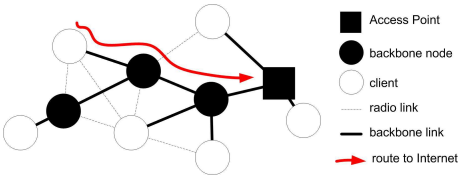


Fig. 1. General behavior of SOMoM

integrate information about the Internet connection parameters (e.g. the IP subnet, the Foreign Agent address,etc.). However, a tree could present problems of robustness: only one radio link failure could potentially disconnect the structure. Consequently, the meshed backbone could be used instead of the tree structure: when a packet must be forwarded by the backbone, it is forwarded through all the radio links between backbone members, and not only through the radio links child→parent. The algorithms for construction and maintenance are proved to be self-stabilizing and highlight in particular properties of stability and robustness.

[1] presents a routing protocol based on this backbone: SOMoM. It creates a multihop cellular network (or *hybrid network*): it integrates an ad hoc bubble in a wired network. The backbone nodes form a distributed cache, and packets are forwarded through the backbone from or to the Internet gateway. The self-organization algorithms already maintain a tree structure (child→parent in the backbone). This allows to create a route toward the root of the backbone, i.e. the Internet gateway. This constitutes a gratuitous proactive route to the Internet, an *upload* route. Inversely, a localization process is initiated when the Internet gateway receives a packet to forward to the ad hoc area, if no route is present in the routing cache for this particular destination. A localization request is flooded but only through the backbone nodes: it reduces largely the overhead. Then, a *download* route is created distributively in the backbone. We adopted a cross-layer approach: the self-organization and routing protocols collaborate in order to flush obsolete information when the backbone topology must be changed. Since the backbone topology is more stable than the radio topology, this improves the routing cache stability. SOMoM increases the route stability and decreases the overhead. The SOMoM structure is depicted in figure 1.

## III. Design and implementation

### A. Software: somomd

We chose to implement SOMoM in a Linux testbed because it is flexible and open-source. Moreover, Linux is extensively used in the scientific community and allows to share easily the different implementations. Additionally, we chose to implement the totality of the protocol in user space: the code is portable, with any Linux kernel, without modification. We implemented a daemon which constructs and maintains the virtual backbone, and the routing protocol SOMoM, described above. This daemon is fully operational: the self-organization is well-maintained and the localization protocol SOMoM functions perfectly. In the following, we detail some key points in our implementation available online [8].

*a) Kernel / user space:* Linux philosophy is to propose a restricted and stable kernel. Programs should be executed in a normal shell, not in kernel mode. However, this property is rarely observed by developers. Indeed, Ad-Hoc Support Library (ASL) [9] allows to implement ad hoc daemon in user space but contains itself kernel modules, for example to modify the routing table. [10] presents 3 methods to implement a routing protocol in MANET. With the Netfilter method, some filters are applied when packets arrive. Some packets can be forwarded to a specific application when they are treated by the IP layer. Alternatively, the daemon could be implemented as a kernel module ([11] is one example of such an implementation). Finally, a third method consists in listening frames in the MAC layer. If an ARP request is generated, it is intercepted and a route discovering is initiated (ARP is in this case mandatory). Kernel implementations seem for us intrusive and present several drawbacks (e.g. kernel instability, implementation changes when the kernel is upgraded). Our daemon is implemented integrally in user-space, and was tested with the kernels 2.6.14 and 2.6.12.

*b) Multi-threads:* The daemon must monitor several tables (neighborhood table, routing table,...) and flush obsolete information. The maintenance is based on triggers and timeouts. So, we chose a multi-threads implementation, each thread monitoring one type of information. Naturally, the implementation must take care of resource conflicts.

*c) Addressing:* Each ad hoc node is configured with a static address with a 32 bits netmask (the *network-netmask*). Therefore, a route in the routing table corresponds to one single destination. All the nodes share a common logical private IP subnet of 24 bits, called the *somom-netmask*. For example, a node can have the IP address 192.168.1.15 with the network netmask 255.255.255.255 and the somom-netmask 255.255.255.0. Other ad-hoc nodes will be configured in the IP range 192.168.1.1-254.

The *somom-netmask* proves its usefulness for the Access Point for example. The AP must distinguish the ad hoc addresses and the Internet addresses. When the destination is known, a 32 bits route exists in the routing table: the packet is sent directly. On the contrary, if the AP has no route, it will verify that the destination IP address owns to the ad hoc area thanks to the *somom-subnet* (obtained from its IP address and the *somom-netmask*).

*d) Routing table:* The kernel routing table was designed for permanent and stable routes: a change is an exception. But ad-hoc routing protocols add and delete continuously the entries in the routing table, and handle timeouts. Hence, an internal routing table maintained by somomd dialogs with the kernel routing table and synchronizes its information (fig. 2). In this way, the kernel routing table is not modified, rendering the source code evolutive.

*e) Protocol stack:* The OSI model requires a strict independence of different layers so that layers interchangeability is facilitated. This independence allows flexibility but decreases performances: no information is shared. Moreover, an information mutualization allows to reduce the overhead. For example, DSR assumes the existence of an API which allows a notification of the MAC layer when an unicast packet is well

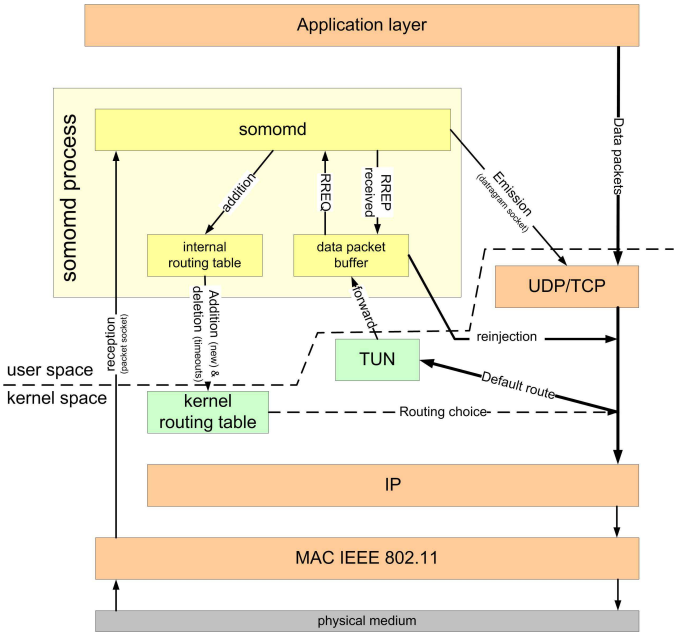


Fig. 2. Software architecture of the protocol

delivered or dropped to improve the route maintenance.

The software architecture is described in figure 2. The daemon is executed above the network layer and uses an UDP socket to send control packets. However, the protocol needs information about all received IP packets. Thus, the daemon implements also a packet socket. This facility allows in Linux to capture all MAC frames (like a promiscuous IP mode). The program extracts IP packets, and information useful to update routing tables for example. Eventually, the packet is forwarded to the routing or self-organization threads if the packet is intended to the UDP port registered by SOMOM. In conclusion, when a packet is forwarded by a node, somomd can update on the fly its routing table.

In particular, each node can add an entry in its routing table for the source of each received packet when the source owns to the *somom-subnet* (computed from the IP address and the *somom-netmask*). In this way, a gratuitous inverse route is learned each time a packet is forwarded by a node.

*f) Reactive behavior:* The code was implemented in the user space. Each ad-hoc node has a default route pointing to its parent in the backbone. If a specific route with a 32 bits prefix corresponds, it will be used. Else, the packet is transmitted through the default route. Hop by hop, the packet reaches the gateway if no route exists for this destination.

When the Access Point receives any packet to forward, it can extract the location of the destination thanks to the *somom-subnet*. If the destination is outside the ad hoc scope, the packet is forwarded through the wired link. If the destination is in the ad hoc area, two cases can occur. If a route (with a 32 bits prefix) exists, the packet is sent through this route. Else, the route toward a client node (a *download* route) must be learned reactively. The process uses the *TUN* facility offered by Linux. In the gateway, a route corresponding to the *somom-subnet* points to the TUN device, and this TUN device points itself to the somomd process. Moreover, when a packet is routed, the linux kernel chooses the route matching with the longest prefix. Hence, the default route is only used in the gateway when no other route exists. Somomd will receive this packet, buffer it in an internal queue, and send a route discovering. In conclusion, we did not modify at all the Linux Kernel.

For a route discovering, the AP sends a `Route Request` in multicast. All the backbone nodes which receive such a packet must forward it in multicast if the destination is not present in their neighborhood table. Else, a `Route Reply` is generated and transmitted through the default route. Hop by hop, the `Route Reply` is forwarded toward the gateway, and creates in each hop an entry in the routing cache. When the AP receives the `Route Reply`, somomd will extract the packets from the internal queue for the corresponding destination and just re-inject them in the normal routing process. The new route will be used. In conclusion, we keep the whole IP routing process without modification

### *B. Node equipment*

The mesh network is constituted by 8 *barebones* (silent and small PCs). Each node is equiped with an hard disk to store an huge quantity of logs in order to extract performance results of the experimentations. Besides, all the static nodes have a wired NIC so that it can be monitored out-of-band, and so that the management traffic does not interfere with experiments. Only one mobile node is introduced in the experiments since to manage mobility during the experiments is a quite difficult task. Other mobile nodes will be introduced in the future to corroborate our current results. However, the radio topology can change because of temporary obstacles (closed door, person in the corridor...). Consequently, even in a topology with only one mobile node, we can test many usual scenarios.

The nodes have all a IEEE 802.11 abg NIC, but we use the 2,4 GHz frequency so that any standard mobile node can be integrated in the hybrid network. Naturally, some tests have been also done with the 5 GHz frequency, and the obtained results corroborate the results exposed in the next section. However, they are not presented here because of lack of space.

In IEEE 802.11, the broadcast frames are always sent with the lowest available bitrate. Oppositely, unicast frames are sent with the bitrate as high as possible. Thus, the radio range of broadcast and unicast frames are different since the modulations differ. This could constitute a severe drawback: `hellos` being broadcasted, a node can choose to forward a packet to an unreachable neighbor. Besides, IEEE 802.11 presents performance anomaly when different bitrates are used [12]. In conclusion, we chose to configure all the clients with a forced 1Mbps bitrate (6 Mbps with IEEE 802.11 a).

### *C. Testbed*

8 nodes were deployed in the lab, constituting a multihop testbed (fig. 3). The topology is a trade-off to test both the network diameter and the network redundancy. One node (represented with a red square in the figure 3) acts as a gateway to the Internet. The gateway implements NAT functions to connect the ad-hoc area to the Internet, using netfiler. The testbed is full-operational and used for a multihop Internet connection in the lab.

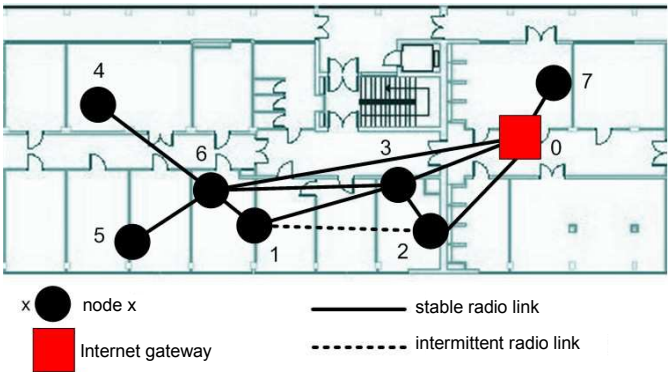


Fig. 3. Map of the testbed

## IV. Experimental results

### A. Self-Organization

In a first time, we evaluated the impact of a topology change on the self-organization: the algorithms must quickly update their information and converge to a legal state. In particular, we measured the impact of the addition/deletion of a node. When a node is inserted in the network, the backbone topology is update in less than 1 second. So, we stressed the protocol in inserting simultaneously two nodes (A and B) to a network of 2 nodes (C and GW). The final network consists in a line $A - B - C - GW$. 2.6 seconds are necessary so that the new nodes determine if they must be backbone nodes or not (tab. I). Additionally, the backbone tree links are valid 0.6 seconds later. Finally, the self-organization structure updated all the information about parents, children in the backbone 4 seconds after the nodes insertion. We say that the backbone is in degraded-mode if the backbone mesh is connected but not the backbone tree (all radio links must be used, not a subset).

| Step | Convergence time (in s) |
|---|---|
| State | 2.6 |
| Valid backbone parent | 3.2 |
| Non degraded mode | 4.0 |

TABLE I
Convergence time when two nodes are inserted simultaneously

Then, we set up a squared network of 4 nodes and shutdown the backbone member neighbor of the gateway: the node 2 hops far from the gateway becomes disconnected, the backbone must be reconstructed, the other neighbor of the gateway must in particular become a backbone node. The disappearance is detected after 4.0 seconds, the timeout of an `hello` (tab. II). The reconnection step lasts on average 1.2 seconds (reconnection packets transmission,...). Finally, the self-organization proves that it functions normally less than one second later. The whole reconnection procedures lasts on average 7 seconds. Naturally, this convergence time can be largely reduced if a more efficient disappearance detection is available (MAC layer notification, very frequent `hellos`...). Other scenarios were studied (for example when 2 nodes disappear simultaneously) and present similar convergence time, but the details are not presented here because of lack of space.

| Step | Convergence time (in s) |
|---|---|
| Disappearance detection | 4.0 |
| Valid backbone parent | 6.2 |
| Non degraded mode | 7 |

TABLE II
Convergence time when a node disappears

### B. Ping

Firstly, we measured the end-to-end delay according to the packet size (fig. 4) from each node to the gateway. We plotted the min/max/average delays. To evaluate the accuracy of the estimation, the errors were plotted for a list of 10 experiments. When the packet size increases, the delay also increases: the transmission requires more time since the radio bandwidth remains fixed. Moreover, the store-and-forward approach increases this effect: the packet must be integrally received before it can be forwarded. Besides, we can remark that average, minimum and maximum delays are very close: the jitter is reduced.

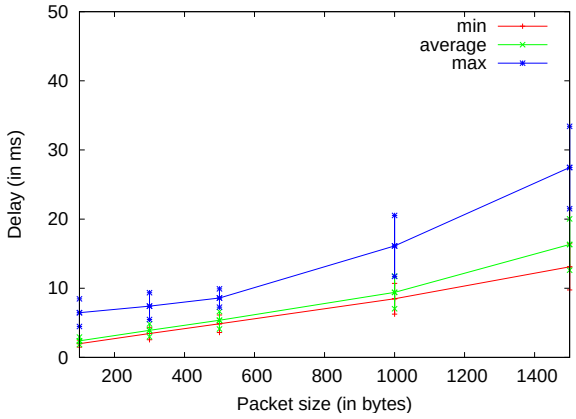


Fig. 4. End-to-end delay according to the packet size

Since the topology is in this part static, we compared the performances of SOMoM with static routing (tab. III). However, a few intermittent radio links impact the performances: routes should be reconfigured dynamically so that the network offers the best performances. We can note that the delay is almost the same with SOMoM and with a static routing whatever the pair in communication is. SOMoM, in spite of a dynamic route discovering, allows to obtain delays similar to a static route configuration. SOMoM presents an high flexibility to topology changes without any impact on the delay. In some cases, we can even note that SOMoM seems more efficient: the protocol adapts its route to the real quality of the radio links, discovering other routes when a radio link becomes weak. We can note that the node 5 which has only weak radio links presents the highest delay.

| Pk Size | Protocol | Source | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 100 | SOMoM | 23 | 4.6 | 5.6 | 5.5 | 15 | 15 | 9.0 |
| | static | 46 | 4.2 | 4.2 | 25 | 52 | 10 | 9.9 |
| 1500 | SOMoM | 597 | 29 | 35 | 34 | 147 | 118 | 144 |
| | static | 649 | 29 | 34 | 64 | 119 | 121 | 82 |

TABLE III
End to end delay in milliseconds with a ping

### C. TCP throughput

Since TCP flows represent the most important part of the Internet traffic, we measured the reachable throughput with iperf (fig. 5) for flows of 8 seconds. We remarked that when a large data flow is forwarded, collisions are created between the different data and control packets. This behavior creates some route instabilities since for example `hello packets` are dropped and a node can consider erroneously that one of its neighbors is dead. A QoS mechanism for IEEE 802.11 would avoid such a behavior. A duration of 8 seconds presents a good trade-off. We can remark that the throughput increases when the packet size is longer. Indeed, some control frames (acks, backoffs...) are required whatever the packet size is. Thus, a long packet size allows to reduce the bandwidth ratio used for control. Moreover, we can distinguish two flow types: the one hop flows offer the highest throughput, and multihop flows propose a lower throughput. However, 2 and 3 hops flows do not differ importantly. A multihop wireless Internet connection seems efficient.

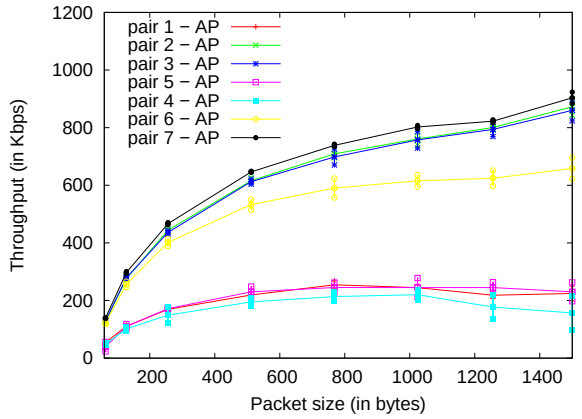


Fig. 5. TCP throughput according to the packet size

We measured the TCP throughput for every source-destination pairs (fig. 6). We can remark that the radio links present very different throughputs: one radio link (the bar of one hop flows) exhibits a throughput of 700 kbps while another one presents only a throughput of 100 kbps because of temporary obstacles. Besides, multihop connections present a lower throughput which is logical since the flow must be forwarded. However, the flows of 2, 3 or 4 hops present similar througputs. Thus, the throughput is quite scalable according to the route length. In other words, SOMoM discovers stable routes, without too frequent breaks.

### D. UDP throughput

Then, we measured the maximum achievable throughput with a UDP flow. Since UDP is not reliable, we assume that a flow is *achievable* if more than 95% of the packets are delivered to the destination. We used here also the iperf tool to measure the UDP throughput. Since we obtained very similar results, the graphs are not reported here. We can just report that an optimal packet size exists to optimize the UDP throughput. When the packet size is large, collisions seem to occur frequently and decrease the global throughput. UDP seems to react less efficiently than TCP to collisions.

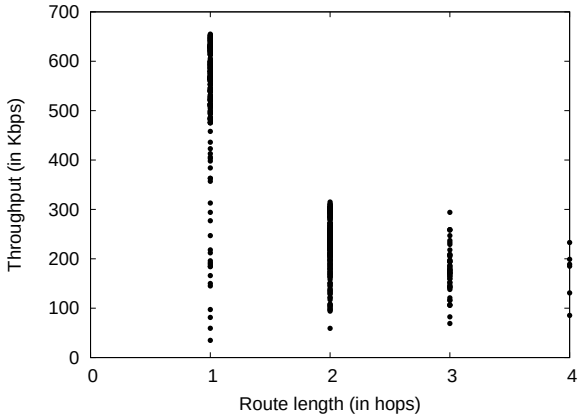


Fig. 6. TCP throughput according to the route length

### E. Route discovering latency

Finally, we measured the delay required to discover a new route from the gateway (tab. IV). For a route of 3 hops, 800 milliseconds are on average required for a ping (route request triggering, potential retransmissions, route reply reception, round trip of the ping). However, a route discovering occurs seldom: when the node initiates itself a communication, an inverse route is automatically created in the cache of the intermediary routers. Moreover, this delay is required only for the first packet of a flow, the route being subsequently dynamically maintained.

| Type | node 4 | node 5 | node 1 |
|---|---|---|---|
| Delay (in ms) | 779 | 894 | 779 |
| Standard deviation | 246 | 384 | 352 |

TABLE IV
ROUND-TRIP-TIME (IN MS) OF A PING WHEN A ROUTE DISCOVERING FROM THE GATEWAY IS REQUIRED

### F. Mobile node

Then, we introduced a mobile node in the hybrid network. This node is moving from one extremity of the network to the Access Point (cf. map in figure 3). We measured the TCP throughput according to the displacement of the node. At the beginning, the mobile node is at the extremity of the network and the radio link is weak: this explains the fluctuating throughput. When the node stops, the throughput is more stable. When the node is changing its parent in the self-organization, the throughput decreases logically: routing reconfigurations are required, disturbing TCP. Finally, when the mobile node is neighbor of the Access Point, the throughput is maximal since it is a single hop flow. SOMoM reacts well to topology changes, updating continuously its knowledge.

## V. CURRENT ISSUES IN TESTBEDS FOR HYBRID NETWORKS

This testbed constitutes a first step toward the improvement of protocols for ad-hoc and hybrid networks, and corroborates

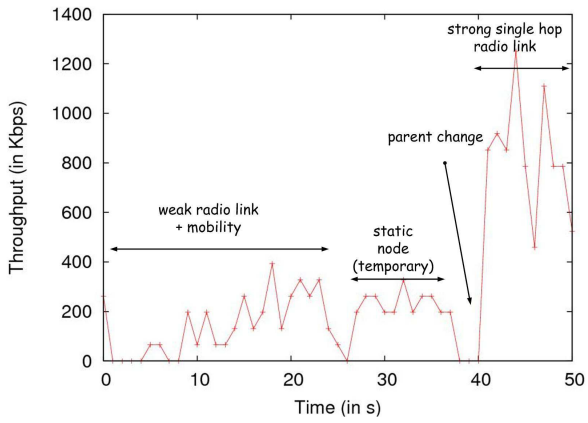


Fig. 7. TCP throughput of a mobile node

the efficiency of protocols based on a self-organization. However, the experiments allowed to exhibit some key problems. Firstly, a real-world radio medium presents severe differences compared to simulations: radio links are heterogeneous, presenting different throughputs. Thus, an ad-hoc protocol must use a metric of link efficiency to improve the global performances, like in [13]. Moreover, the timescale of route changes must be adjusted to the timescale of a radio link. For example, it is useless to change a route after 3 seconds if a radio link is changing every second. Besides, some radio links are instable, their radio range being not binary: some long links allow to deliver at most $x\%$ of the packets. If a `hello` is received, the radio link is considered valid although it presents a poor throughput. Thus, we modified SOMoM so that a radio link is considered valid if at least two consecutive `hellos` were received. Although the testbed is constituted by similar nodes, some unidirectional radio links can appear because of antenna orientation, bad connectors.... Hence, unidirectional and bidirectional links must be distinguished by the protocol (like SOMoM does). The environment highly influences the radio topology. In other words, the Unit Disk Graphs are a bad model of ad-hoc networks: two nodes can be near without having a radio link with each other. The dimensioning of a mesh network must in consequence carefully be conceived. Finally, tests are time-consuming. Automatized tools must be deployed and clocks must be synchronized. However, the control traffic for the measurements must avoid to disturb the experiments. Thus, a out-of-band management should be available (via wired connections, or different wireless NIC and radio channels)

Besides, we point out several key problematic properties of IEEE 802.11. They must be addressed to set up an efficient radio multihop network. IEEE 802.11 presents poor performances in an ad hoc network, under-estimating the bandwidth. A new MAC layer must consequently be proposed. Moreover, IEEE 802.11 does not offer priorities among flows. Thus, an heavy data flow will collide with control packets, disturbing the normal functions of a protocol, as mentioned above. Routes will be broken, creating instabilities in the throughputs Besides, IEEE 802.11 does not present the same rate in unicast and broadcast (as exhibited in these experiments and earlier in [14]). Thus, control packets sent in broadcast are transmitted farther. A node can consider a node neighbor although it is not able to send or receive a data packet in unicast

## VI. Conclusion & Future Work

This article presented the deployment of a testbed to measure the performances of hybrid networks in a real environment. In particular, we implemented a self-organization protocol and the routing protocol benefiting from this scheme, SOMoM. We described the complete software architecture: this explanations represent a guide to implement protocols in a flexible manner in a Linux testbed. The integration of ad hoc networks in the Internet is fully operational, offering a spontaneous multihop extension of IEEE 802.11 networks. The performance evaluation demonstrates the relevance of a self-organization scheme and of SOMoM. Besides, we detail some specific issues of IEEE 802.11 in multihop radio testbeds, presenting problems of stability and QoS. A new MAC layer particularly adapted to multihop wireless networks must be proposed to optimize the performances.

As a future work, we plan to deal with multi-interfaces hosts: Bluetooth, WIFI, Ethernet interfaces must be seamlessly integrated, forming a wide ad-hoc network, offering a transparent connection to the Internet across multi technology links. Besides, more nodes must be introduced to test the scalability.

## References

[1] F. Theoleyre and F. Valois, "Mobility management in multihops wireless access networks," in *PWC*. Colmar, France: IFIP, 2005.

[2] D. A. Maltz, J. Broch, and D. B. Johnson, "Experiences designing and building a multi-hop wireless ad hoc network testbed," School of Computer Science, Carnegie Mellon University, Technical Report CMU-CS-99-116, March 1999.

[3] J. T. Kaba and D. R. Raichle, "Testbed on a desktop: strategies and techniques to support multi-hop manet routing protocol development," in *MOBIHOC*. Long Beach, USA: IEEE, 2001.

[4] D. Gray, Robert S.and Kotz, C. Newport, N. Dubrovsky, A. Fiske, J. Liu, C. Masone, S. McGrath, and Y. Yuan, "Outdoor experimental comparison of four ad hoc routing algorithms," in *MSWIM*. Venice, Italy: ACM, 2004.

[5] J. Bicket, D. Aguayo, S. Biswas, and R. Morris, "Architecture and evaluation of an unplanned 802.11b mesh network," in *MOBICOM*. Cologne, Germany: ACM, 2005.

[6] W. Kiess and M. Mauve, "A survey on real-world implementations of mobile ad-hoc networksnext term," *Ad Hoc Networks*, in press, 2006.

[7] F. Theoleyre and F. Valois, "A virtual structure for mobility management in hybrid networks," in *WCNC*. Atlanta, USA: IEEE, 2004.

[8] somomd, "Available on: http://sourceforge.net/projects/somom."

[9] V. Kawadia, Y. Zhang, and B. Gupta, "System services for implementing ad-hoc routing: Architecture, implementation and experiences," in *MOBISYS*. San Francisco, USA: ACM, 2003.

[10] I. D. Chakeres and E. M. Belding-Royer, "AODV routing protocol implementation design," in *IWWAN*, Tokyo, Japan, 2004, pp. 698–703.

[11] L. Klein-Berndt and et.al, "Kernel AODV implementation," http://w3.antd.nist.gov/wctg/aodv_kernel/.

[12] M. Heusse, F. Rousseau, G. Berger-Sabbatel, and A. Duda, "Performance anomaly of 802.11b," in *INFOCOM*. San Francisco, USA: IEEE, April 2003.

[13] D. S. J. De Couto, D. Aguayo, B. A. Chambers, and R. Morris, "Performance of multihop wireless networks: shortest path is not enough," *ACM SIGCOMM Computer Communication Review*, vol. 33, no. 1, pp. 83–88, 2003.

[14] E. Lundgren, E. Nordstro, and C. Tschudin, "Coping with communication gray zones in ieee 802.11b based ad hoc networks," in *WOWMOM*. Atlanta, USA: ACM, September 2002.