

# Definition and Evaluation of a Trellis Structure for Self-Organized Networks

Julien Ridoux  
LIP6 - UPMC

8, rue du C. Scott. Paris, FRANCE  
julien.ridoux@lip6.fr

Anne Fladenmuller  
LIP6 - UPMC

8, rue du C. Scott. Paris, FRANCE  
anne.fladenmuller@lip6.fr

Yannis Viniotis  
ECE Dpt - NCSU - Box 7911  
Raleigh, NC 27695, USA  
candice@ncsu.edu

**Abstract**—Self Organized Networks (SONs) are a general description of autonomous networks, of which Ad Hoc and Sensor networks are special cases. Lack of infrastructure, limited resources and dynamic nature of nodes are some of the main characteristics of such networks. We had previously proposed a novel clustering architecture for SONs, that utilizes a fixed description of cluster routing tables. The architecture uses trellis graphs, a concept borrowed from the domain of source coding, to provide addressing/locating and routing capabilities for the SON. The choice of a trellis graph (and its parameters) to represent a particular SON is not unique; moreover, as expected, it introduces overhead. In this paper we focus our study on a thorough examination of the effects of this choice on the control and data plane operations of the network. In a nutshell, we show that increasing the trellis size reduces control plane overhead significantly; on the other hand, increasing the trellis size has a mixed effect on data plane performance.

## I. INTRODUCTION

Self-Organized Networks (SONs) are a new paradigm in networking as they provide an innovative way for users to build networks at will. Ad Hoc or Sensor networks are both representative examples of SONs. The basic property is that SONs do not rely on any infrastructure and each node might equally be in charge of the routing process. Another one is that the network topology is not necessarily known in advance when the nodes are forming the network. Finally nodes can be mobile and/or transient (e.g., when using energy saving modes, or in the case of nodes/links failures).

Due to such inherent properties, SONs face unique technical challenges. Wireless SONs, for example, must perform network operations with limited resources (bandwidth, battery). This has been a focus of several studies, that proposed ways to minimize the cost of routing. Another challenge concerns the scalability of the proposed approaches. As the network grows, the routing tables defined by numerous approaches will grow as well, causing more processing, storage and data exchange to maintain the tables.

In [1] and [2] we provided a new addressing and consequently a new routing scheme to address the challenges we mentioned. Details of our proposed solution have already been presented in [1] and [2]; only a brief summary will be presented here. The main focus of this paper is on how the essential parameters needed to correctly design our proposal can be selected.

From a high-level point of view, we propose to build a *hierarchical* clustering of SONs based on *regular* structures. The regularity of the structures is essential in defining clusters precisely. Our approach imposes the organization of the nodes within the clusters by the use of a predefined routing table layout and fixes the size of a cluster. We choose to use *trellis graphs* as the basic pattern of the routing table layout because of their redundancy properties and their description as a Finite State Machine. With these tools, we are able to define a recursive addressing scheme and a consequent routing strategy.

In terms of addressing, our proposal ensures correlation between the address of a node and its location in the network. This configuration and address allocation process is done independently for each cluster, and hence can be performed in a distributed, scalable manner. Our approach does not impose any limit on the network size; simply, the trellis structure increases as more nodes join the network. As a drawback, building an optimized, trellis-based structure (*i.e.*, achieving the minimum number of clusters) is an NP-Complete problem. We have proposed a heuristic algorithm that builds the structure efficiently [2]. In terms of routing, our approach provides a straightforward equivalence between the address of a node and the route to reach it. The size of the routing table is fixed, providing the routing mechanism with good scalability properties. Our approach provides a built-in multi-path scheme that is suboptimal in the data plane, but has good robustness properties.

The rest of the paper is organized as follows. In Section II, we describe prior work about addressing and routing in Self-Organized Networks. In Section III, we describe the main features of our approach to build the cluster structure and summarize the main principles of the associated routing strategy. Finally, in Section IV, we provide an extensive study on how one can choose the parameters of the proper basic trellis graph and evaluate the impact of such choices on the control and data plane operations of the network.

## II. PRIOR WORK

In this section we focus our analysis of existing work on solutions proposed in the following research directions: IP-based solutions for Mobile Ad Hoc networks, Geographic routing approaches and cluster-based ones. We exclude the Peer-To-Peer approaches in the following section as their use

at the application layer does not fit our needs based on the knowledge of the network topology.

Numerous Ad-Hoc network protocols ([3],[4],[5],[6]), based on IP addressing have been designed to allow a dynamic auto-configuration of nodes and an implied routing protocol definition. These protocols provide a straightforward compatibility with the existing IP network, but do not decorrelate identification and location of nodes. The IP address used is then limited to contain solely the identification information. References [7], [8] provide a distributed address allocation but use a Duplicate Address Detection (DAD) mechanism and the implicit flooding to ensure the uniqueness of each IP address. In the context of Ad Hoc and Sensor networks, the DAD mechanism is not totally reliable and still does not eliminate the possibility of IP address conflicts, due to the transient presence of nodes. References [5],[6],[3] provide distributed routing algorithms but generate a high overhead of control messages, especially when nodes are highly mobile. Under these conditions more frequent flooding is required for DAD and route discovery that may prevent them from scaling. Concerning routing itself, these approaches face a complex route discovery process. Without any structure available, they rely on improved flooding mechanisms to retrieve a route to a destination, and few of them have extensions to handle multi-path. This process can lead to broadcast storms, and thus may exhibit low efficiency. In general, these IP-based approaches are well suited for a SON context when few node faults, and no network splits or mergers occur, but will perform with more overhead otherwise.

Geographic routing [9] implicitly decorrelates the node identities and their geographic locations. By assuming nodes to have a Universal Identifier (UI), geographic approaches get rid of possible identity conflicts and propose a distributed and more manageable geographic addressing space. Geographic addressing and routing possess excellent scalability properties and routes are computed easily thanks to the geographic position associated with nodes. Decorrelating identity and location implies the need for such routing mechanisms to retrieve the geographic location of a destination before being able to contact it. In this context, the Grid Location Service (GLS [10]) is an example of the use of Virtual Regular Structures (VRS) used for localization purposes. In order to allow the nodes' location information to be distributed among the nodes, GLS imposes a VRS known a priori by all nodes and divides the geographic space into a hierarchy of grids. With this predefined quad-tree structure, GLS provides a deterministic method for the nodes to store and retrieve location information. Routing approaches coupled with GLS achieve most of the SON requirements we discussed. However, reliability depends on the node density of the network. When geographic zones of the network have a low node density, routing and location information management can be difficult to realize. This is the main reason why we do not consider the geographic information to be well suited for a "good" location management scheme and instead propose a topological description.

Finally, approaches based on cluster definition have been proposed in the context of Ad Hoc and Sensor networks [11]. The cluster approach has been mainly developed out of scalability concerns, in order to avoid having to flood the entire network. Many different methods have been proposed to realize a clustering of the network and we can not enumerate all of them here. Their main properties concern the size of the cluster defined, some topology-aware metric and a usually two-level hierarchy differentiating basic nodes from cluster heads.

All these proposals have good points or properties; we tried to include them in our approach, while proposing a new point of view on the problem and its solution.

### III. CLUSTERING STRUCTURE

Our approach realizes a clustering of the network into sets of nodes, in order to ensure addressing/location and routing functions. We define the set of nodes forming a cluster and their corresponding cluster heads. Contrary to existing proposals, our approach defines and imposes constraints on the organization of nodes within the cluster. Each cluster has the *same* maximum number of nodes and imposes the *same* organization amongst them. In the rest of this section, we briefly describe the construction process of our approach because of space limitations; more information can be found in [1] and [2].

#### A. Network Organization

In traditional IP networks, the routing table is usually described as a tree, whose shape and depth depends on the network connectivity. In our approach, we adopt a novel point of view: we enforce a specific routing table layout that each node will use, prior to any network operations. We chose trellis graphs as the routing table layout for several reasons. They are *regular* structures of *fixed size*, repeating a graph pattern that can be defined with little information. The fixed size of the routing table layout defines the maximum number of entries/nodes in each cluster. Moreover, the use of a regular structure is well suited for the environment of SONs where no particular property can be inferred from the physical topology. Introducing regular patterns provides clusters with desirable built-in multi-path properties. These multi-path properties provide a nice basis to set up a robust routing scheme, that can cope with SON peculiarities, such as sleeping modes, wireless link failures, and mobility as we will detail in section III-C.

1) *Trellis-Based Routing Tables*: A trellis is a connected graph, repeating a set of vertices, all of which have the same degree. Traditionally used for signal encoding and decoding operations, a trellis can also be described as a Finite State Machine (FSM). In the following we limit our description of trellis graphs to these two descriptions; more information can be found in [12]. As an example of these two descriptions, figure 1 represents a trellis of size 4 and its corresponding representation as a FSM. The vertices of the trellis, labeled by binary values, represent the different FSM states. The edges of

the trellis represent the FSM transitions. The transitions of the FSM are labeled by binary values indicating the input given to the FSM. The sequence of states resulting of a run of the FSM corresponds to a path in the trellis. As an example, from state 00, the FSM changes to state 10 if the new input value to the FSM is 1 and stays in state 00 if the input value is 0. The same example can be seen on the corresponding trellis.

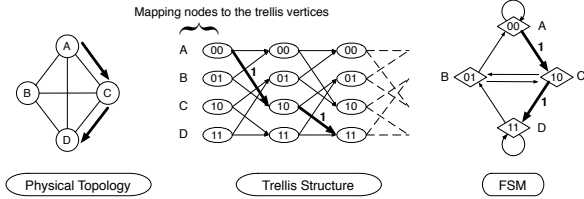


Fig. 1. Node Mapping to a trellis of size 4.

As nodes join the network, they exchange topology neighborhood information to combine and define their own instance of the routing table (valid within a cluster). This combination operation is realized by mapping the edges of the regular structure (defining the routing table) to existing physical links. Each edge of the routing structure corresponds then to a physical link while the opposite is not necessarily true.

In figure 1 we illustrate the mapping of a physical topology composed of 4 nodes A, B, C and D (left graph) to a trellis of 4 vertices (middle graph). This mapping associates each node of the topology to at least one vertex of the trellis. As an example, node A in the physical topology shown in figure 1 is associated to the vertex of the trellis labeled 00. Once the mapping is realized, the trellis structure defines a virtual topology with routes between nodes (right graph). The labels of the trellis vertices define then the addresses of nodes within the cluster, shared by all the cluster nodes.

Note that *numerous* trellis can be generated, with different node degrees, introducing more or less redundancy and consequently, a larger or smaller number of possible paths between two distinct vertices. How a choice between various alternatives can be made, to address networking concerns, is the subject of section IV.

2) *Recursive Organization*: All clusters are connected in a tree structure built recursively. The cluster labels reflect this virtual tree structure. In figure 2 we arbitrarily chose a maximum trellis size of 4, to map this physical topology to the virtual one shown at the top. Particular nodes, called the Trellis Heads (indicated in grey color), are present in different successive levels of the recursive hierarchy and define redundant paths along the structure. A cluster element is thus either a node or a cluster. For instance, in figure 2, cluster 1 is composed of nodes C and D (representing cluster 10) and nodes A and B (representing cluster 11). The example shows that the mapping between nodes and vertices of the trellis within a cluster is not a bijective operation. Note that in the cluster labeled 110, node G is mapped to two different entries of the routing table (00 and 11), adding even more redundancy.

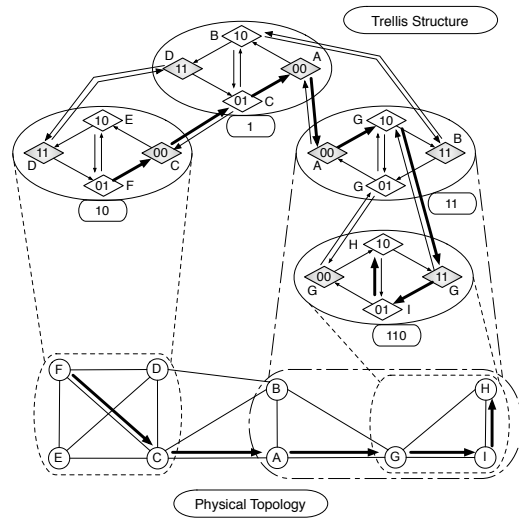


Fig. 2. Hierarchical Clustering Illustration.

### B. Destination Location Service

As soon as a node joins the network it is included into the cluster structure. The cluster's label and the node's mapping to a trellis vertex represent the new node address in the virtual structure. This address indicates the location of the node in the structure and gives a straightforward routing decision at each cluster on the path. The association between this address and the node's unique identifier (IP address for example) is propagated and stored by the trellis heads at the higher levels of the structure, solving then the location service issues met in geographic routing.

When a source node needs to reach any destination in the network, it sends a request propagated specifically to the trellis heads of higher levels. This location mechanism is similar to the GLS approach [10] and ensures that the source node will retrieve the location information of the destination.

### C. Routing and Data Forwarding

Once the source node retrieves the information of the destination, the forwarding operations based on the produced trellis structure are realized along two policies. The first one, called intra-trellis routing, is based on the FSM description and is illustrated in figure 1 (boldface paths). It takes advantage of the trellis graph structure to facilitate data forwarding operations.

The second policy, inter-trellis routing, is realized between each cluster the packet has to go through. Recall that the clusters in the structure are labeled as a virtual tree structure as they are created. The inter-trellis routing uses the destination's information present in the packet header. The destination's cluster label is used to determine the next cluster in the path. The cluster inter-connection is maintained thanks to the hierarchical virtual structure. In the example presented in figure 2, node F sends packets to node H, along the paths shown in boldface. In the cluster labeled 10, F uses intra-trellis routing to forward the packet to node C. Node C, as

a trellis head, forwards the packet to the next cluster on the path. This operation is repeated until reaching the cluster of the destination, and finally node H.

The intra-trellis routing we describe provides a default route in the structure that may not be the optimal path within the cluster. Since a relay node knows the complete path the packet will follow in a cluster, it can use its physical neighborhood information to define shortcuts in the default route and then optimize the path length. In section IV, we will refer only to this routing improvement when examining simulation results. Deploying inter-trellis routing improvement is still under work at the moment, and is not a part of the present study.

#### *D. Maintenance Operations*

Since our approach is based on the introduction of a structure in the network, node join and leave events require some maintenance operations. These events may have an impact on the coherence of the structure and on the forwarding process.

When a new node joins the network, it is inserted into a cluster thanks to a local interaction with its physical neighbors. If a cluster of one of the new node's neighbors is not "full" and if the new node connectivity allows it, then the node is inserted into the cluster. Otherwise, if no possibility is found, based on existing clusters, a new trellis cluster is formed using already configured neighbors as trellis heads. Once the node is inserted into a cluster, it triggers a registration in the structure as mentioned in section III-B. When a node leaves the network, the vertices of the trellis it was associated to are reallocated to the remaining nodes of the cluster. These two operations are limited to the clusters impacted by the join/leave operation and do not need to be propagated in the entire network.

When a node joins the network, it handles the forwarding operations associated to the vertices of its trellis. The new node may handle forwarding operations that were previously handled by one of its neighbors. It is important to notice that our routing scheme does not rely on which precise node is forwarding the packets on the path to the destination, but requires only that a node be associated to the correct vertex in the trellis structure. This constraint is handled by construction. When a node that was executing forwarding operations leaves the structure, alternate paths can be easily found within the cluster thanks to the redundancy introduced by the trellis pattern, until the trellis vertices it was mapped to are allocated to other nodes in the cluster. Again, the departure of a node does not require that the route be rebuilt entirely; it must be updated only locally, by forwarding nodes. The overhead introduced by such operations is the subject of the "control plane" evaluation in section IV.

#### *E. Properties of our approach*

Since we did not describe extensively the construction process of the trellis structure and the associated routing mechanisms, we present here only a brief summary of properties, advantages and drawbacks of our proposal. Our approach decorrelates the identity of a node from its location, as in geographic or Peer-To-Peer approaches. At the same time, we

introduce an equivalence between the address of a node and its location, as in the ideal IP address description scheme, to improve scalability. We introduce a recursive definition of the clusters borrowing the name service concepts from GLS. The recursive definition of the addressing space allows us to handle any physical topology, regardless of the geographical density of nodes in a particular area.

In terms of addressing, the trellis approach brings back correlation between the address of a node and its location in the network. The address of a node in a cluster, defined by the mapping of its identifier to a vertex of a trellis, represents a part of its connectivity to its direct neighborhood and its location relative to the other nodes forming the cluster. The address of a node is then relative to its topological location in a cluster and in the entire network. It is worth noticing that the resulting construction of the clusters and their instance of the routing table is dependent on the physical topology of the network and on the sequence of node arrivals.

The control plane operations required to build the addressing space are realized on a local basis and are fully distributed. Our approach avoids flooding messages in the entire network. The information is shared by nodes of a cluster at each level of recursion and does not have to be disseminated blindly. As we presented in subsection III-A, a node can be mapped to different addresses in a trellis, improving the robustness of the approach. Another property we did not detail so far, concerns the mobility of nodes participating in a self-organized network. Thanks to the recursive description of the address of each node, our approach can deal with network splits and mergers without having to rebuild the entire structure.

The main drawback of our proposal in terms of addressing stems from the structured approach itself. In a dynamic environment, our proposal adds a cost to the protocol in order to maintain the structure coherent. Nevertheless, we designed our structure while having this in mind. First of all, the redundancy we introduced in the structure avoids having to trigger a maintenance operation each time a node moves or leaves. The structure is robust in time, until too many physical topology changes force it to update. Since the structure is defined recursively, changes in the topology will not impact the entire structure and will remain localized to only a single branch of the hierarchy.

In terms of routing, the trellis structure defines a built-in multipath scheme thanks to the natural redundancy characteristics of the trellis graphs. This multi-path scheme is the basis of the robustness of our approach. From this multi-path scheme, our approach designs an implicit equivalence between addresses of nodes and routes to reach them. Our approach does not require route discovery processes: as long as a source node knows the address of the destination, it is able to send its packets to the network. We define then an implicit loose source routing thanks to the description of the routing table as a Finite State Machine.

The main drawback of our approach in terms of routing concerns the route length in the data plane. The default routing scheme we propose is suboptimal, since it uses only intra-

trellis optimization. Some inter-trellis routing optimization has to be designed to provide at least one route closer to the optimal one. Nevertheless, the set of multiple paths defined (that can not all be as good as the optimal one) provides robustness and a strong basis for load balancing.

#### F. Choice of the virtual structure

In our approach we impose two strong constraints on the cluster organization: the maximum size of the cluster, and the default forwarding paths within a cluster. It is possible to generate a large number of trellis of different size and edge patterns. The impact of the edge pattern is difficult to quantify. This mapping depends on the physical topology and the nodes' arrival sequence (as briefly shown in figures 1 and 2). We keep the evaluation of this parameter for further studies.

Contrary to other clustering approaches, the trellis structure constraints the maximum cluster size. We consider that this parameter would probably have a major impact on the performance of our approach as it impacts the total number of clusters formed and their coverage of the network. We aim to find which size of the cluster would be the most proper. We choose two metrics to evaluate the impact of the trellis size parameter: (1) The overhead in the control plane (*i.e.*, the cost to maintain the structure coherent) ; our results are summarized in table III. (2) The overhead in the data plane (*i.e.*, the path length between any pair of nodes) ; our results are presented in figures 3 and 4.

### IV. RESULTS AND INTERPRETATION

Our goal is to evaluate the influence of the size of the cluster on the two metrics we have just introduced, and, if possible, to decide which size could be the "best". For this purpose, we will study trellis graphs of 4 different sizes: 4, 8, 16, and 32 vertices. We chose not to consider trellis of higher size, to maintain the locality property of our approach in terms of address allocation.

As we presented in figures 1 and 2, the characteristics of the physical topology impact the construction of the clusters through the mapping realized. We then observe the impact of the trellis size parameter toward 3 categories of topologies: a clique graph, grids and random topologies. Clique and grids are not realistic as a SON topology; we study them on a comparison basis to improve our understanding of the impact of the cluster size.

The results presented in this section are based on a C++ simulator specifically written for this proposal. It is used to compute the control and data plane metrics proposed. The simulator considers wireless nodes with a transmission range of 250m, a value commonly used for simulations of Ad Hoc networks. The physical topologies we use for this study are then realized by specific placement of nodes and their resulting neighborhoods.

#### A. Physical Topologies

As the characteristics of the physical topology are a matter of interest, we precisely define some statistical metrics in

order to compare the different kinds of topology we use in the following (characteristics of the topologies studied are presented in table I; they are mainly used in section IV-C). The first metric we consider is the *Link Density*: it is defined as the ratio between the total number of links existing in the network and the maximum number of links that could exist. The second metric is the *Average Distance* of a graph, defined as the average value of the shortest paths between any two nodes of the topology. The third metric we use to describe a topology is the *Clustering* of the graph: this is the probability that two neighbors of a node are also neighbors. Finally we consider the *Average Degree* and the *Distribution of Degree* of nodes that counts the number of neighbors of each node.

The first topology we consider is a clique graph, in which each node has all others as neighbors. For comparison to the other topologies, and when not explicitly mentioned, we will consider a clique graph of 625 nodes.

The second category of graphs we consider are grids. Grids are interesting for our study as regular structures with a very small variance in their node degree distribution: except for nodes on the border of the grid, all the other nodes have the same degree. In the following we refer to two different grids, each composed of 625 nodes (25x25). The first grid is referred as "Grid 4" and is composed of nodes having a degree equal to 4. The second grid, called "Grid 8", is a more connected one, where nodes have a degree equal to 8. In both Grid 4 and Grid 8, we deterministically placed the nodes on each grid by numbering them from the center nodes. Because of this deterministic placement, we can study the impact of the trellis size with only one simulation run.

The last kind of topologies we have considered are random topologies: for a given geographical density (*i.e.*, given number of nodes present in a unit square) we have placed 625 nodes in a square area in a uniform fashion. The dimension of the square area in which nodes are placed depends on the geographical density chosen. We produced 3 sets of random topologies of densities 35, 45 and 55 nodes per km<sup>2</sup>. Since these topologies are produced randomly, the results presented here correspond to 20 runs, using the same conditions for each topology density. Table I presents the values of the statistical metrics we use to describe the different topologies.

TABLE I  
STATISTICAL PROPERTIES OF TOPOLOGIES

Topology	Link Density	Avg Distance	Clustering	Avg Degree
Clique	1	1	1	624
Grid 4	0.014	15.33	0	3.9
Grid 8	0.028	10.74	0.44	7.49
Random 35	0.010	14.20	0.591	6.52
Random 45	0.013	11.23	0.594	8.30
Random 55	0.016	9.85	0.603	11.10

#### B. Trellis Structure & Control Plane

1) *Scalability*: As we mentioned in the previous section, we limit our study to networks composed of 625 nodes. This

limit is a choice we made and is not driven by any limitation of our approach in terms of scalability in the trellis structure size produced. In order to justify this point, we varied the number of nodes from 50 to 1000 and used every size of trellis graph we study. As we have observed, except for topologies of less than 200 nodes, the number of trellis per node is almost constant, regardless of the size of the topology, showing the good scalable properties of the clustering structure construction. The size of the network is then not really important when evaluating the control plane operations. We choose a size of 625, a value that seems relevant to observe the data plane performance within a reasonable processing time.

2) *Structure size*: We observe now the size of the overall structure produced, depending on the characteristics of the topology and size of the trellis graphs chosen. Obviously, as the maximum size of the cluster increases, the number of clusters decreases. However, the characteristics of the topology make this relation be a bit more complex. Table II presents the number of recursive clusters produced for each topology and each trellis graph size. For example, observing the topologies Random 35 and Random 55, for the same increase of the trellis graph size, the number of recursive clusters produced does not decrease at the same rate.

By comparing tables I and II we are able to identify some trends in the way the characteristics of the physical topology influence the trellis structure construction. The link density indicates how the physical constraints of the topologies affect mapping the nodes to the trellis graph. When the link density is high, the total number of links increases and allows more possibilities to build the recursive cluster structure. The clique topology is a “lower bound” case, in which no constraints are present and the best trellis structure can be built. These results are consistent with the evaluation of the construction heuristic presented in [1].

The average node degree does not seem to have a strong influence at first sight. Indeed the trellis vertices degree is fixed and equal to 2, in the pattern we use. So having more neighbors does not impact a node already configured in the structure. Nevertheless, a higher node degree indicates that new nodes joining the structure have more possible neighbors. More neighbors implies that these new nodes possibly have more distinct clusters they can choose to join, leading to a more compact structure. In the case of random topologies, the link density and the average node degree increase together.

The clustering of the physical structure does not seem to have any influence on the construction of the trellis structure. The fact that two neighbors of a node are also neighbors, only impacts the structure based on trellis of size 4, where nodes of the cluster are at most 2-hop neighbors. For trellis graphs of higher size, the clustering is not a very accurate characteristic and its influence is difficult to determine.

The Average Distance of the physical topology has no influence on the trellis structure construction and will be used when evaluating the performance of the approach in the data plane.

In conclusion, having larger trellis sizes allows the designer to build a smaller number of hierarchical trellis structures, even though the relation between the size of the trellis and the resulting structure is affected by the characteristics of the physical topology.

TABLE II  
TRELLIS STRUCTURE SIZE

Topology	#Trellis 4	#Trellis 8	#Trellis 16	#Trellis 32
Clique	312	104	45	21
Grid 4	623	312	203	154
Grid 8	401	182	108	75
Random 35	403.48	198.05	125.75	102.10
Random 45	396.50	185.10	121.79	95.85
Random 55	406.05	176.80	110.85	86.20

3) *Maintenance Cost Function*: Our goal here is to provide a simple evaluation of the cost incurred by maintenance operations in the control plane. For this purpose, we define a cost function (see equation 1), that will approximate the cost of the largest maintenance operation on the structure. This cost function will be used to compare the different trellis graphs sizes.

We define  $T$  as the trellis structure our heuristic can build.  $T$  is composed of  $N_T$  recursive clusters, each of them of size  $s_T = 2^l$  ( $l$  varying from 2 to 5 in the work presented here). As we mentioned in the previous section, the size of the resulting trellis structure depends on the trellis graph size and the characteristics of the physical topology.

In order to measure the efficiency of the construction, we define  $r_t$  as the node redundancy ratio of a recursive trellis  $t$ . Obviously the set of all  $t$  corresponds to the trellis structure  $T$ . We let  $n_t$  denote the number of distinct physical nodes gathered in  $t$ .  $r_t$  corresponds to the proportion of vertices of the trellis  $t$  mapped to distinct nodes and is defined as  $r_t = \frac{n_t}{s_T}$ . In order to compare all the structures produced, we define  $R_T$  as the average value of  $r_t$  for all the trellis of  $T$  ( $R_T$  is measured based on the simulations and the obtained values are presented in table III). In each trellis,  $r_t$  is an indicator of the construction of the clusters and depends on the neighborhood constraints amongst nodes.  $r_t$  is an indicator of the mapping of the nodes to the vertices of a trellis, and is then dependent on the connectivity constraints imposed by the trellis and its size.  $r_t$  depends on the characteristics of the topology and the trellis graph size.

Now that we have precisely introduced the notations and some relations between the metrics used, we define  $m$  as a maintenance message corresponding to a node join into a cluster.  $m$  will serve as the unit to measure the cost of maintenance operations. This cost we want to look at will be composed of the number of times a message  $m$  has to be relayed to reach the correct number of destinations. We denote by  $C_T$  the cost of the maintenance operation of a trellis structure  $T$ . In general, the cost function  $C_T$  is the total number of relayed messages within the structure to realize the maintenance operation. Evaluating the exact expression

of  $C_T$  is complex since it depends again on the size of the trellis graph used, the number of clusters produced and the characteristics of the physical topology. We then propose a pessimistic approximation of the cost by a function  $\widehat{C}_T$ .

$$C_T \approx \widehat{C}_T = N_T [R_T l + (1 - R_T)(l - 1)] \quad (1)$$

In the following we detail how we derive  $\widehat{C}_T$ . The values of  $N_T$  and  $R_T$  are measured from simulation results to compute the values of the bound  $\widehat{C}_T$  presented in table III.

To define the cost function, we consider two ways of relaying the message  $m$  in the structure. We first consider the number of times a message  $m$  has to be relayed within a cluster to reach all nodes participating in the cluster. We consider the “worst” case for the transmission in a cluster of size  $s_T = 2^l$  where each vertex of the trellis graph has been associated to a distinct node of the network. Thanks to the properties of the trellis graph and the mapping realized, and by taking advantage of the wireless medium, the message has to be relayed exactly  $l$  times in order to reach each node of the cluster. In the case where the number of distinct nodes in the cluster is lower than  $s_T$ , the number of times the message will have to be relayed is bounded by  $l$  and 0 (when only one node is present in the trellis cluster) and depends on the node triggering the forwarding of a maintenance message. The value  $l$  is the cost for the nodes that are associated to a unique vertex of the trellis graph. For the other nodes, we provide an upper bound to the cost by fixing it to  $l - 1$  in all other cases. This allows us to give an overall upper bound to the cost of the maintenance in a trellis by weighting this two costs by  $r_t$  thanks to the heuristic used to set up the clusters. An upper bound cost of the maintenance operation within a cluster can then be expressed as:  $r_t l + (1 - r_t)(l - 1)$ . Replacing  $r_t$  by its average value  $R_t$  gives us the bracketed term in equation 1. It is important to notice that the bigger the trellis graphs (*i.e.*, the bigger  $l$  is), the looser this upper bound is. To derive equation 1, we suppose as a worst case, that the maintenance operation involves all  $N_T$  recursive clusters composing the structure<sup>1</sup>.

We computed the empirical approximation of the cost function for each topology and each of the trellis graph sizes we consider. Results on this computation are presented in table III. From this table, we observe that for each topology, the cost of the maintenance operation decreases as the size of the trellis graph used increases. This behavior is particularly true for the clique topologies. In the case of grids and random topologies, the cost does not decrease at the same rate. The cost difference between trellis of size 16 and 32 is not very significant compared to the other topologies.

In summary, similar to the discussion about the trellis structure size, our main finding is that control plane overhead can be reduced by choosing trellises of higher size.

### C. Trellis Structure & Data Plane

In this section, we observe the influence of the trellis size on the data plane. We still use the three different physical

<sup>1</sup>We did not include maintenance costs due to node leave events, since they are much smaller than the maintenance costs corresponding to a node join.

TABLE III  
COST MAINTENANCE RESULTS

Topology	Trellis 4		Trellis 8		Trellis 16		Trellis 32	
	$R_T$	$\widehat{C}_T$	$R_T$	$\widehat{C}_T$	$R_T$	$\widehat{C}_T$	$R_T$	$\widehat{C}_T$
Clique	0.99	620	0.99	310	0.99	179	0.99	104
Grid 4	0.75	1090	0.50	780	0.32	673	0.19	645
Grid 8	0.89	757	0.68	487	0.49	376	0.32	324
Random 35	0.86	749	0.63	520	0.42	427	0.25	433
Random 45	0.87	740	0.66	492	0.44	416	0.26	404
Random 55	0.88	763	0.69	473	0.48	382	0.29	368

topologies and the same simulation environment; we focus on the length of the routing path as our main metric of evaluation. We calculate the empirical PDF (Probability Density Function) of the path length and the corresponding path stretch CDF (Cumulative Density Function), for different trellis sizes on a given topology. The stretch is defined as the ratio of the routing path length divided by the physical shortest path length.

TABLE IV  
OPTIMIZED PATH LENGTH

Trel. Size	Random 35		Random 45		Random 55		Grid 4		Grid 8	
	Avg	Std Dev	Avg	Std Dev	Avg	Std Dev	Avg	Std Dev	Avg	Std Dev
4	26.6	14.2	24.8	12.4	22.2	10.2	23.1	8.4	18.9	7.4
8	27.1	14.3	22.3	10.9	19.7	9.5	35.7	13.0	18.7	6.9
16	26.0	13.6	21.3	10.3	19.4	9.9	33.3	12.6	18.9	7.1
32	28.6	15.5	22.6	11.2	19.9	9.9	37.4	15.3	22.1	9.1

1) *Impact of the optimization:* In section III-C we described the construction of the default route within a cluster and its optimization by taking into account the physical neighborhood of each relay node. In order to clarify the analysis, we start by observing the impact of this optimization on the path length of the topology Random 35. Figure 3(a) presents the distribution of path length for each trellis size, without any optimization. This PDF plot shows then the default route built with the trellis structure. Figure 3(a) indicates the distribution of physical shortest path length and the default routes obtained on this topology, for each structure built with a different trellis graph size. A first point to notice is that the PDF plots of the default path length for different trellis graphs size are very close to each other and barely distinguishable. From this first observation, we conclude that the size of the trellis graph used does not impact the size of the default route produced. This observation is reinforced by the CDF plots in figure 3(b). The stretch distributions of the default routes seem to be equivalent, except for the stretch distribution produced by trellis graphs of size 4 that is slightly under the other plots. We conclude that the size of the trellis has a very small impact on the default route length produced.

Figure 3(c) shows stretch CDF for optimized routes on the topology Random 35. Again, we notice that the path length distributions are barely distinguishable. The simple intra-cluster optimization we propose with the routing mechanism

has a comparable impact on the trellis structures, regardless of the trellis graph size.

This optimization, independent of the size of the trellis graph (and, therefore, the mapping of the physical topology), greatly improves the stretch of routes. In figure 3(b), around 30% of the routes have a stretch smaller than 2, while in figure 3(c) almost 70% of the paths have a stretch smaller than 2. This optimization is then efficient to produce paths of (close to) shortest length, even if some additional optimization can be introduced between different clusters. Nevertheless, this extra optimization will not be required for the purpose of comparing the impact of the size of the trellis graph. In the following, we will then only focus on path lengths resulting from the intra-cluster optimization we described in section III-C.

In order to facilitate understanding of the figures describing optimized routes, table IV gathers information on the optimized path length distributions: the average path length and the corresponding standard deviation.

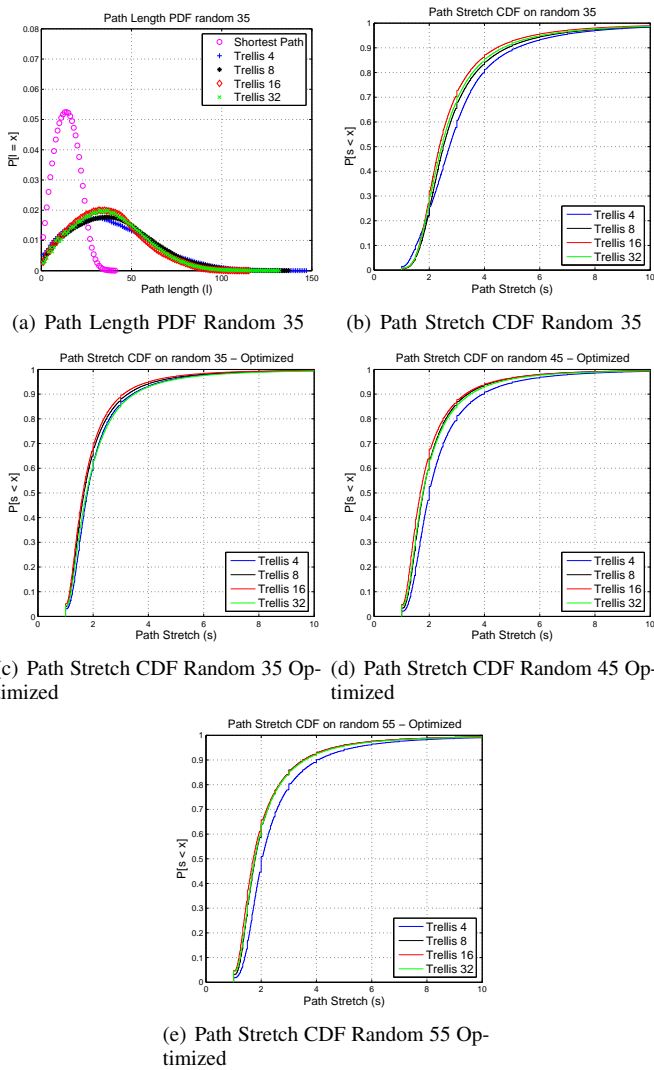


Fig. 3. Path Stretch CDF on Random

2) *Impact of geographical density increase:* Considering only the optimized path lengths produced by the trellis structure, we observe their behavior as the geographical density of nodes increases. Figures 3(c), 3(d) and 3(e) show this behavior and corresponding stretch, based on the 3 topologies. Again, we observe that the distributions of path lengths based on trellis graphs of different sizes remain close to each other. Nevertheless, in figure 3(e) we can observe that the structure built with trellis graphs of size 4 performs slightly worse than the other ones. This is strictly due to the physical characteristics of the topology. When the geographical density increases, as it is the case when comparing the topology Random 55 to Random 35, the average node degree and the link density increase as well. This allows structures built with trellis graphs of higher size to fit the physical topology better than trellis of lower size. This is the trend we can observe by comparing the 3 random topologies, where the path length distribution and stretch produced by structure of different trellis size are slightly scattering.

In figures 3(c), 3(d) and 3(e) we can observe that only the structures based on trellis of higher sizes, namely 16 and 32, manage to keep around 70% of the paths with a stretch smaller than 2. This is particularly interesting, as the physical shortest paths tend to have smaller length. On the one hand, the good point is that the trellis-based structure manages to fit the evolution of the physical topology. On the other hand, all trellis graphs are not equivalent. When the link density and average node degree remains low, the size of the trellis does not make any difference. When the network tends to be denser, larger trellises perform better. Even though this is a small difference, trellis graphs of higher sizes as 16 or 32 perform better than the smaller ones.

3) *Impact of node degree variability:* We observed that the link density and the average node degree of the physical topology have a small influence on the performance of the data plane, when comparing structures set up with different trellis sizes. In order to understand this better, we look at the results produced by the grid topology: if we do not take into account the nodes placed on each side of the grids, all the nodes have exactly the same degree. This situation is different from the random topologies where the variance of the distribution equals the mean.

Figure 4 shows the results of the optimization on the routing paths on the grid topologies. Concerning the topology grid 4, the optimization makes the distributions tend to the same one (figure 4(b), except for the structure based on trellis graphs of size 4). In this particular case, the constraints of the topology on this particular trellis are so strong during the construction of the structure, that seldomly more than two nodes are inserted into the same cluster. Since grid 4 has a clustering null, the path in the trellis with the optimization comes closer to the physical shortest path than the structures based on larger trellises. These ones gather more nodes, but have difficulties in optimizing the paths within the bigger clusters. In the case of the topology grid 8 (in figure 4), the optimization tends to produce close distributions.

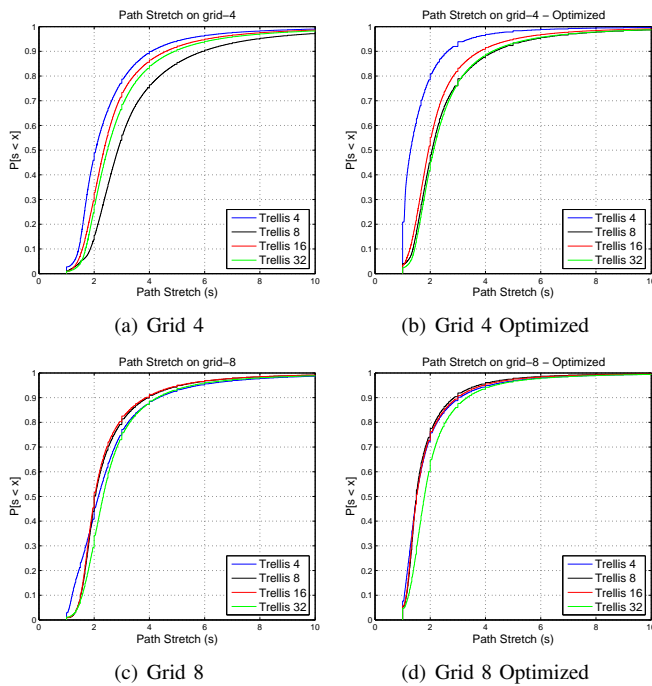


Fig. 4. Path stretch CDF on grids

Finally, when comparing the stretch CDF of grid topologies to random topologies, we observe that the performance on the grid is equivalent or better. That could be useful to steer our approach more specifically toward applications that require or authorize a deterministic placement of nodes. We also notice that it seems more difficult to have a clear answer on what is the optimum trellis size on such regular topologies.

## V. CONCLUSION AND FURTHER WORK

We have proposed the use of a trellis structure as a promising routing table layout to organize clusters in a Self-Organizing Network recursively. We have briefly discussed architectural properties of this proposal, in terms of multi-path, routing, addressing/location and scalability. The work presented in this paper was focused mainly on studying how physical topologies and the parameters of the trellis structure (in particular its size) affect control and data operations of a Self-Organized Network. We have identified the need for trellis graphs to be relatively large (16 or 32 vertices) to maintain small overhead in the control plane while ensuring good performance in the data plane.

We tried to provide an extensive study of the trellis size, a parameter that could impact the performance of our approach in the control and data planes. In the control plane, the use of trellis graphs of larger size (such as 16 or 32) seems to be a good choice to decrease the cost induced by the structure maintenance operations. In the data plane, the size of the trellis graph does not deteriorate the path stretch significantly. This result was not the intuition we had before doing this study. In the data plane, a slight trend to prefer trellis of higher size appears and again trellis of size 16 or 32 can fit most of the situations.

Apart from these results, we have seen that the performance of our approach can be improved in some particular situations requiring a deterministic placement of nodes.

Future work will include: (i) an evaluation of how dynamic node characteristics impact the choice of the trellis size, (ii) developing algorithms to perform inter-trellis cluster routing optimizations, (iii) performance comparison to other proposals, and, (iv) a more detailed robustness analysis.

## REFERENCES

- [1] J.Ridoux, A.Fladenmuller, Y.Viniotis and K.Salamatian, "Trellis-based virtual regular addressing structures in self-organized networks," in *Proc. of IFIP Networking2005, Waterloo, Canada*, May 2005, pp. 511–522.
- [2] J.Ridoux, A.Fladenmuller and Y.Viniotis, "Virtual trellis routing: how regular structures can ease network layer operations," in *Proc. of Med-Hoc-Net 2005, Ile de Porquerolles, France*, June 2005.
- [3] D.B.Johnson and D.A.Maltz, "Dynamic source routing in ad hoc wireless networks," *Mobile Computing, Kluwer Academic Publishers*, vol. 353, 1996.
- [4] C.E.Perkins and P.Bhagwat, "Highly dynamic destination-sequenced distance-vector routing (dsv) for mobile computers," in *ACM SIGCOMM'94 Conference on Communications Architectures, Protocols and Applications*, 1994, pp. 234–244.
- [5] C.E.Perkins, E.M.Royer and S.Das, "Ad hoc on demand distance vector (aodv) routing," IETF, RFC 3561, July 2003.
- [6] C.Adjih, T.Clausen, P.Jacquet, A.Laouiti, P.Minet, P.Muhlethaler, A.Qayyum and L.Viennot, "Optimized link state routing protocol," IETF, RFC 3626, October 2003.
- [7] S.Nesargi and R.Prakash, "Manetconf: Configuration of hosts in a mobile ad hoc network," in *Proc. of IEEE INFOCOM*, June 23-27 2002.
- [8] H.Zhou, L.M.Ni and M.W.Mutka, "Prophet address allocation for large scale manets," in *Proc. of IEEE INFOCOM*, April 2003.
- [9] J. M.Mauve and H.Hartenstein, "A survey on position-based routing in mobile ad hoc networks," *IEEE Network Magazine*, vol. 6, no. 15, pp. 30–39, November 2001.
- [10] J.Li, J.Jannotti, D.De Couto, D.Karger and R.Morris, "A scalable location service for geographic ad hoc routing," in *Proc. of ACM MOBICOM*, August 2000, pp. 120–130.
- [11] J.N.Al-Karaki and A.E. Kamal, "Routing techniques in wireless sensor networks: A survey," *IEEE Wireless Communications*, vol. 11, no. 6, pp. 6–28, 2005.
- [12] S.Lin, D.J.Costello, *Error Control Coding: Fundamental and Applications*, ser. Electrical Engineering Series. Prentice-Hall, 1983.