Trellis-Based Virtual Regular Addressing Structures in Self-Organized Networks

Julien Ridoux[1], Anne Fladenmuller[1], Yannis Viniotis[2], and Kavé Salamatian[1]

[1] LIP6 - UPMC, 8, rue du C. Scott, 75015 Paris, France
julien.ridoux@lip6.fr, anne.fladenmuller@lip6.fr, kave.salamatian@lip6.fr
[2] ECE Department - NCSU, Box 7911, Raleigh, NC 27695, USA
candice@eos.ncsu.edu

Abstract. Self-Organized Networks are multi-hop networks that do not rely on any infrastructure. Moreover, their topology is supposed to be flat since all nodes are equally in charge of address allocation and routing functions. Their time-changing topology breaks the association between Identification and Location that is present in the IP address. Moreover, mobility can not be handled easily when the addressing space is based on a tree description. In this paper we propose the use of Virtual Regular Structures to provide desired properties for Self-Organized Networks. Our approach opens new ways to build addressing spaces and enables an implementation based on trellis graphs. We propose a construction heuristic and evaluate its performance via simulations. We also analyze the robustness of the proposed approach with regards to mobility.
Key words: Self-Organized Networks, Regular Structures, Dynamic Address Allocation

1 Introduction

Self-Organized Networks (SON) represent a network model proposed to support spontaneous and multi-hop networking without relying on any precise and *a priori* infrastructure. SONs are a general description unifying well-known research fields such as Ad-Hoc or Hybrid[3] networks, Peer-To-Peer systems and Sensor networks. A SON is a spontaneous multi-hop network whose time changing topology depends on node arrivals in an environment without infrastructure.

In SONs, routing, addressing, location management, name resolution, all are non-trivial issues because of the node mobility. We argue in this paper that the choice of the addressing scheme has a serious impact on all these issues. An address can uniquely identify a node, inform on its localization, or it can also represent a combination of both. When it only identifies a host, the system has to ensure the uniqueness of each address and mobility has no influence on the address value. On the other hand, if position information is included in the address, the address has to be modified each time a node moves or appears

[3] We define hybrid networks as Ad-Hoc networks that can obtain an Internet connectivity with the aid of some of their nodes.

in the network. This increases the cost of mechanisms used to disseminate or retrieve the association between a node's identifier and its location. The address allocation and routing problems can then be seen as a generic minimum cost query problem to execute on an infrastructure-less environment; nodes query addresses and routes.

A large suite of Ad-Hoc network protocols based on IP addressing have been designed to allow a dynamic auto-configuration of nodes and a routing protocol definition ([1],[2],[3],[4],[5],[6]). While providing straightforward compatibility with wired IP networks, this approaches inherits the hierarchical description of IP addresses applied to a flat topology. These proposals address the query problem with a flooding mechanism: improved flooding requests for route discovery and a Duplicate Address Detection (DAD) mechanism for address allocation. Geographical routing [7] answers the route query problem thanks to the nodes geographic coordinates. Their identification is decorrelated from their location, implying the need of a positioning system and a location mechanism such as GLS (Grid Location Service) [8], to realize the address queries. Peer-To-Peer systems [9] define a virtual addressing space based on Distributed Hash Tables to realize the correspondence between the node identifier and its location in the virtual space. The routing query problem is solved by the underlying layer. The approach in [10] pushes the P2P concepts into the routing layer but its addressing space (based on a tree structure) copes poorly with the dynamic nature of SONs. To avoid flooding, GLS and the Peer-To-Peer approaches introduce a virtual structure that decreases the cost of the addressing query problem. The study done in [11] shows that structured networks are helpful to introduce a coherent addressing scheme and improve routing and maintenance performance. We claim that the use of a virtual structure will give "good" topological properties to SONs by decorrelating the node identifier from its location.

Our proposal provides a topological Virtual Regular Structure (VRS) to map the addressing space, fitting the geographical node placement. We propose trellis graphs as the basic element to implement a VRS in SONs. Trellises can be used for both creating the address space and finding routing paths. Thanks to their redundant structure, they provide an implicit multi-path construction and are robust toward frequent node arrival and departures.

The key point of our proposal is to design an addressing space that is set up in a "constructive" and local manner. This set up avoids the need of a global view of the network, difficult to obtain in a SON. Such a construction adapts the size of the addressing space to the number of nodes present in the SON while providing high scalability by definition. Our proposal avoids address allocation conflicts, without widespread flooding, allowing our solution to scale large networks by construction. In terms of routing, our virtual structure includes redundancy that provides an implicit multi-path definition that is an essential property for SONs.

In the rest of the paper, section 2 describes the general use of regular structures and the trellis implementation. Sections 3 details the addressing space construction. Section 4 contains analysis in terms of robustness of our approach in the presence of mobility. Section 5 presents simulations showing the feasibility of constructing a regular VRS.

2 Generic Virtual Structure

2.1 Generic Description

We propose the use of VRS in SONs to decrease the cost of the solution to the generic query problem described. This regular structure has to possess properties that facilitate both addressing and routing. This regular structure is repeated to integrate all the nodes of a SON. In order to allow a communication between these regular structures, they have to be connected to each other. To keep the same properties induced by the regular structure, we propose to connect them in a recursive manner. Some of the nodes act at different levels of recursion and provide the connectivity between those levels. These nodes are called "Structure Heads", represented in grey and black color in figure 1(a).

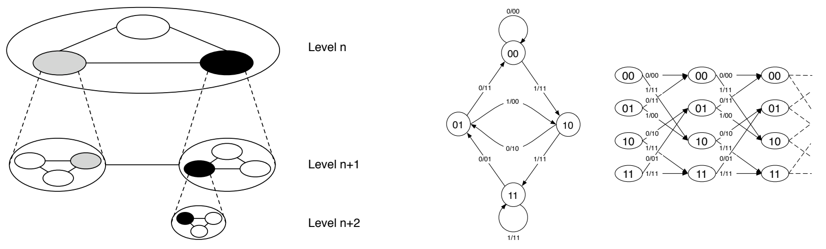
To be able to route in such a VRS, each regular structure must possess an address that identifies it and gives the recursion level in which it is present. Moreover, each node of a regular structure must possess at least one address relative to the structure and all nodes in a regular structure must be able to communicate to each other, through a path in the physical layout.

The address of a destination contains the path to reach it in the whole virtual structure. This path is composed of the list of Structure Heads of each regular structure a packet has to go through. Locally to a structure, the routing decision is taken compared to the address of the next Structure Head. The knowledge of which regular structure to forward packets to is given by the address identifying each regular structure. In order to realize this routing mechanism, each link present in the virtual structure must correspond to a physical one.

2.2 Realization based on Trellis Graphs

We chose to use trellis graphs to implement the general concepts we just described. Trellis graphs allow robustness toward mobility of nodes thanks to the redundancy they imply. Their algebraic definition allows an easy and natural multi-path routing. To the best of our knowledge, trellis graphs have been used only in [12] at the networking level. Their application in providing a VRS is novel. A trellis graph can be generated by a Convolutional Encoder [13]. Figure 1(b) represents such a trellis graph and its representation as a Finite State Machine (FSM).

The FSM states are labeled by binary values indicating the current bits stored in the encoder memory. The transitions are labeled by binary values (*e.g.*, 0/11) indicating the entry given to the FSM (0) and the corresponding output code (11). The same information is represented on the corresponding trellis. From state 00, the FSM changes to state 10 if the new input value is 1 and stays in state 00 if the input value is 0. Words are encoded by reading them in Least Significant Bits order.



(a) A recursive regular structure. (b) FSM and Trellis representation.

Fig. 1. Virtual Regular Structures.

A trellis is a connected graph, repeating a set of vertices, all having the same degree. The vertices of the trellis represent the different FSM states and their edges the FSM transitions. The representation of the FSM state changes for a given input sequence corresponds to a path in the trellis. In the following we use the FSM and the trellis illustrated by figure 1(b) to detail the mechanism used for addressing and routing.

3 Virtual Address Space based on Trellises

In order to precisely describe the way we organize the Virtual Regular Structure, we state some hypotheses first. We consider a Self-Organized Network built from scratch, where node arrivals are sequential. The physical topology created by this arrival process can not be predicted and we assume no particular property on it. We suppose that no infrastructure is present either, to help the addressing or routing mechanisms[4]. We make the hypothesis that each node joining the network possesses a unique Universal Identifier (UI). The definition of this identifier is outside the scope of this paper; the reader can refer to the large amount of literature in different fields such as Peer-To-Peer systems [9]. To allow compatibility with the existing IP network, an IP address could be used as the node UI, as long it is guaranteed to be unique. Our approach can then be implemented as a routing daemon in any Ad Hoc routing protocol, providing full compatibility with existing systems.

3.1 Description of a Single Trellis

The vertices of a trellis graph represent the location of nodes within the virtual structure we want to set up. The FSM associated to the trellis is known *a priori* by all the nodes composing the network. The state of the FSM associated to a node is called its Local Relative Address (LRA). Since the identifier and location

[4] An existing infrastructure can be present and would ease the construction of the VRS in a transparent way. Here we consider the general case without any infrastructure.

of a node are decorrelated, we need to introduce an address Association Table to map the node UI to its Local Relative Address. Figure 2 gives two simple physical network topologies to illustrate this mapping where capital letters represent the nodes UI. The topology of figure 2(a) is composed of four nodes each associated to one of the FSM states.

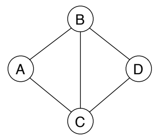
In figure 2(b), since the number of physical nodes is less than the number of states of the FSM, node E is represented twice, *i.e.*, E is given two addresses in the regular structure. As shown in this example, each trellis possesses a given number of addresses and one or several of these addresses are allocated to a physical node. The resulting Association Table representing neighbor nodes depends on their arrivals. Different node arrival sequences, resulting in the same physical topology could produce different node UI to FSM state mappings, and different numbers of Local Relative Addresses of a node. The Association Table creation is shortly described in section 3.3. In the examples given in figure 2, the paths to nodes are their LRA. If node A needs to communicate with node D, it will use D's LRA (11). By construction, each edge of the trellis corresponds to an existing physical link. A can then find a path in the trellis to reach D.

3.2 Spanning the entire Network

Recursion is introduced by creating trellises of trellises. This recursive construction is our proposed solution to span the entire network by repeating the same regular structure. Figure 3 shows trellis graphs T_3, T_4 and T_2 mapping respectively the nodes $\{A, B, C, D\}$, $\{E, F, G, H\}$ and $\{I, J, K\}$.

In order to allow all the nodes of the network to communicate, the trellis graphs T_2, T_3 and T_4 are interconnected in a recursive manner. Figure 3 presents a VRS of two levels of recursion, where T_1 associates T_3 to its states 00 and 10, while it associates T_4 to its states 01 and 11. T_0 and T_1 can then be described as "trellises of trellis graphs".

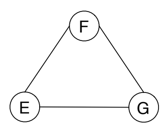
The VRS structure we describe aims to provide both an addressing space and a routing protocol. But trellis graphs belong to the virtual space. To communicate from one trellis to another, the communication has to be ensured by the physical nodes interconnecting the two trellises. To realize all the operations our mechanism needs, trellis graphs are represented at any level of recursion by some of the physical nodes that permit this interconnection. The nodes associated to the lowest and highest value of the FSM states are chosen to represent



Node UI	Node LRA
A	00
B	01
C	10
D	11

(a) 4 nodes mapping.

Node UI	Node LRA
E	00
E	01
G	10
F	11



(b) 3 nodes mapping.

Fig. 2. Basic Physical Topologies and Association Tables.

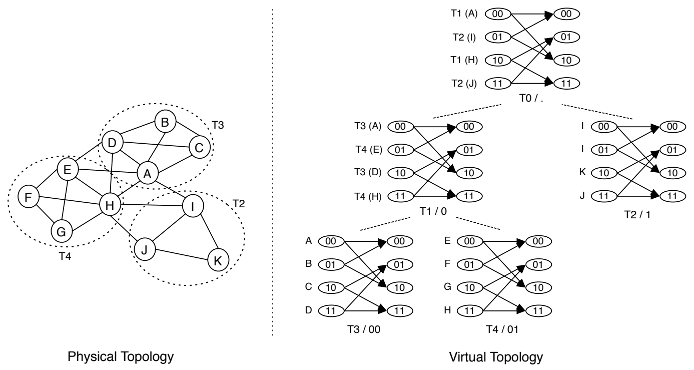


Fig. 3. Spanning the Network.

their trellis in higher recursive levels. These states are selected because they own a loop edge in the FSM making easy to represent the recursive interconnection. We name these two connection nodes the Trellis Heads. In figure 3, T_3 is represented in T_1 by its Trellis Heads A and D. In the same manner, T_4 is represented by nodes E and H in T_1.

As we mentioned in section 2, we need to define a Trellis Prefix (TP) to identify each regular structure. By definition each trellis of size 2^L combines $2^{(L-1)}$ sub-trellis. Each trellis T attributes a prefix p written as a bit string of size $2^{(L-1)}$ to the nodes representing a common sub-trellis. p is the recursive concatenation of T's TP with the bit string attributed by T to a sub-trellis. The highest trellis in the recursive hierarchy possesses a prefix of size 0. T_4's TP (01) in figure 3 illustrates this concatenation of T_1's TP and T_4's prefix in T_1.

Once the trellis structure is established, all the states of each trellis composing the VRS are associated to a physical node. It is then possible for a node to retrieve the path to any destination in the VRS. This path, called the Relative Address (RA), is a bit string composed of the concatenation of Local Relative Addresses. For each trellis a packet goes through to reach the destination, one LRA is concatenated to the Relative Address. The LRA added in each trellis is the LRA of the last node the packet goes through before being forwarded in another trellis. The RA is the sequence of "exit vertices" of each trellis to go through to reach the destination. Since the RA corresponds to a path, a destination RA is different for two distinct source nodes.

As an example, consider K's Relative Address in figure 3. Let suppose that F and J need to reach K. For J, the case is trivial. J and K belong to the same trellis, they share the same Association Table, and J knows K's RA as 10. From F's point of view, K's RA will be 00.11.11.10. That means E's LRA in T_4 (00), H's LRA in T_1 (11), J's LRA in T_0 (11), K's LRA in T_2 (10). It is important to notice, that this is not the only possible RA for K from F's point of view. Since we introduce redundancy for multi-path properties, and because the RA is bound to the path, 11.00.01.10 would have been another possible path.

3.3 Heuristic for Trellis Construction

Establishing an optimized trellis-based VRS, as we just described, is intuitively the biggest difficulty of our approach. In a more formal manner, the construction of the optimal trellis-based VRS is an NP-Complete problem as we have shown in [14]. The optimality criterion defined involves a minimum number of trellis and a minimum number of recursive levels to cover the whole topology. Since finding an optimal trellis VRS is an NP-Complete problem for a given topology, we have proposed a heuristic to set it up, based on an incremental arrival of nodes. In this way, our heuristic builds the top trellis of the VRS first and adds new, lower level trellises as nodes join the SON. When a node joins a network, it tries to join the existing VRS by being inserted in an existing trellis. If no possible position is available, because of physical connectivity constraints, the arriving node creates a sub-trellis with its neighbor. This process ensures that the heuristic is able to configure any new node joining the network. As we will describe later, our proposal allows an implicit merging of trellises when nodes do not appear sequentially. More details about this heuristic are given in [14].

3.4 Packet Forwarding

As nodes join the network, they register recursively their positions in the virtual topology until reaching the highest level trellis. This registration is similar to the one proposed in GLS in the case of geographic routing. The route request from a source node S is propagated to higher levels of hierarchy until reaching the first node L possessing an entry for the requested destination D. By concatenating the path from S to L and from L to D, the source node retrieves a complete path to the destination.

Each forwarding node realizes a routing decision based on the address of the destination in the topology (present in the packet header), the FSM defining the trellis and its own location in the VRS. As the address of the destination nodes represents a sequence of states in the FSM, this process gives a direct routing decision on a per-trellis basis, making it possible to forward the packets along the structure. Because of space limitation, we do not describe here the mechanisms used for packet forwarding in detail. A complete description can be found in [14].

4 Robustness Analysis

Mobility is arguably the biggest challenge SONs have to face in regards to addressing and routing. Our approach decorrelates node identifier and position, which provides the basis to handle mobility. In order to position our work toward the different mobility aspects, we define four mobility categories. The first one is the join and leave processes of nodes. The second kind of mobility, “slow individual mobility”, is characterized by a continuous node mobility, where the node movement can be tracked gradually by its neighbors. The third aspect can be defined as “fast individual mobility” and does not allow the tracking of the mobile node. The node is present continuously in the network but its neighborhood changes completely between two given instants. The last category of mobility we consider here is group mobility. This type of mobility leads to network splits and mergers, that are the aspects any addressing space (except the geographic one) has difficulty to handle. Combining these different kinds of mobility creates numerous scenarios that can not be described here.

4.1 Slow Mobility

Because of the effect they have on our scheme, we combine the join and leave process with the individual slow mobility of nodes. The join process of a node is handle by the construction process of the VRS. When a node moves or leaves, it produces broken links in the trellis graphs it was present. Because of the introduction of redundancy, the structure remains robust and is able to find alternative paths to route packets that were intended to go through the missing node. Since a trellis is a representation of a Convolutional Encoder, the analogy with error recovery is straightforward. While the number of active links remains over the capability threshold of the trellis, there exists a path (obviously longer than the shortest one) to reach an existing destination. When the number of active routes falls under the threshold, the trellis structure does not allow routing and needs to be reconstructed.

The reconstruction of a trellis is a local procedure. This is arguably the best advantage toward mobility, since a trellis reconstruction will not have an impact on the entire network. When the number of routes of a trellis falls under the threshold, the nodes remaining in the trellis restart a bootstrap process as if they were just joining the network. Since their UI remains the same, the protocol is able to guarantee reception of all packets they have to receive, at a cost of some retransmissions.

4.2 Fast Mobility

Fast Individual Mobility of a node changes its neighborhood and therefore its topology knowledge changes very fast. In order to be able to handle this kind of behavior, our approach must be able to configure trellis graphs, register addresses and route packets at a higher speed than the node movement. Due to the complexity of the trellis set up, and the possibility for having continuous trellis reconfigurations, we do not claim that our trellis-based VRS is able to handle such node behavior. Nevertheless, we think that our approach will cope with most realistic situations.

4.3 Group Mobility

Because our VRS is based on a neighbor node construction, two new-formed network partitions, that result from a network split, remain coherent. They form two independent recursive subtrees of the VRS. For each network partition, some nodes representing the missing trellis graphs would have disappeared in different recursion levels. This situation is handled as a successive sequence of node departures. The trellis graphs forming the new border with the other partition may have to be reconfigured. Again, the split operation is handled locally and does not influence the entire network. If a higher level trellis completely disappears, the prefixes of the remaining trellis graphs have to be updated by removing the bit string corresponding to its Trellis Prefix.

A network merger can be detected when, for example, two configured nodes M and N meet and the request for a route to each other does not give any result for each of them. A network merger has no influence on the addressing space, or on the routing topology thanks to the recursive property of our structure. In other words, the VRS of a prior network is placed "under" the other one.

In order to guarantee that the whole VRS remains coherent, M has to become a part of N's trellis. At the same time, in order to allow packets to be forwarded between N and M's networks, M has to be present at the top trellis of its partition. These conditions imply that, M chooses to join N's trellis because its VRS contains fewer recursive levels of trellis graphs. If M is a trellis head present at the highest level of hierarchy, it joins N's trellis, and the network merger procedure is finished. If this is not the case, M starts a trellis head permutation procedure to become a trellis head present at the highest level of recursivity.

Our approach takes advantage of the redundant and recursive properties of the VRS to handle broken routes, slow node mobility and group mobility. Even though we did not describe the mechanisms involved precisely, because of paper length limitation, we show the advantages of such properties for SONs.

5 Evaluation and Simulations

It is always possible to build the VRS containing all the network nodes with recursive trellises. Figure 4 is an example of a resulting trellis-based VRS construction. Since such construction is an NP-Complete problem for an existing topology, it is important to simplify the construction as much as possible, while maintaining acceptable performance in terms of computation load on the nodes for the routing, minimization of the information storage required for our VRS or matching of the physical minimum route and the path in the VRS to the destination.

Since we use only local information to optimize routing, we expect the trellis-based shortest path to always be in the same order of magnitude as the physical shortest path. We computed the difference of the path length between the physical shortest path and the path produced by the trellis-based VRS. We do not present the obtained simulation results in this paper, as the number of simulation runs was not sufficient to have a real statistical meaning. So far these first encouraging results give us the indication that the path length does not increase exponentially.

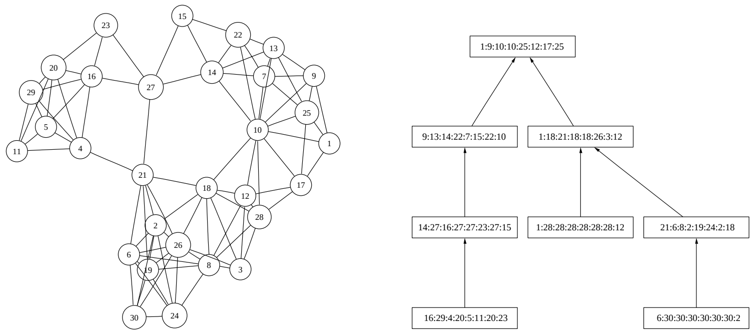


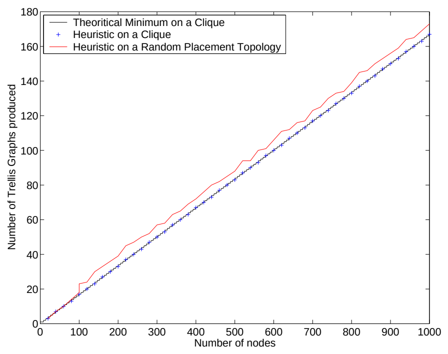
Fig. 4. 30 nodes Random Placement Topology and its VRS

The computation required to route packets along the VRS mostly depends on the length of the path between nodes. The longer the path in the VRS, the more processing time is required to compute it, since more nodes are involved in encoding bit strings representing addresses. Statistically, it seems intuitive that the depth of the VRS will have an impact on the average length of the path between nodes. The depth of the VRS can give us some insight on the performance in terms of average routing processing load for a given topology.

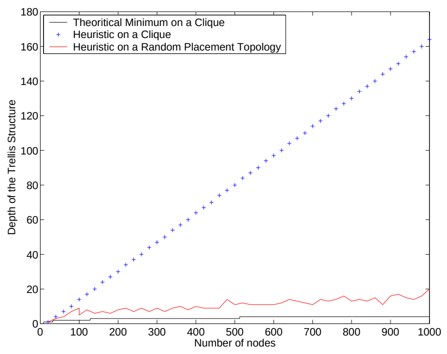
Each trellis stores information on each association (LRA, UID) of its own trellis and of all the trellises of longer prefixes below it. The size of the trellis is fixed, and several LRAs can be associated to one UID. So, in order to optimize the amount of information stored in the VRS, we need to minimize the number of trellises and to create the shortest structure possible. This is what our heuristic tends to do, by limiting the creation of new trellises [14]. Figure 4 shows the resulting VRS thanks to our heuristic on a random placement topology.

As a result of this analysis, we can deduce that the depth of the virtual structure and the number of trellises has an impact on the performance of our approach. Although our heuristic tends to minimize both metrics, it can not be guaranteed they will both be optimized. One major characteristic of our proposal is that, although we are sure a VRS can be built, we can not predict its resulting shape. As a matter of fact, for a given topology, several VRS can be built, depending on the node arrival sequence: the number of trellises as well as the maximum number of recursive trellises (depth of the VRS) can differ. It is thus important to understand which parameter will impact performance the most.

In order to evaluate the construction heuristic we proposed, we generated two kinds of topologies. One is a fully-connected topology, where all nodes are 1-hop neighbors. The second one is a randomly placed node topology on a square area of size 1000x1000 meters. Each wireless node possesses a transmission range of 250 meters, a widely used value for simulators (as in NS-2 for example). In these simulations, we studied the construction of the VRS that is the core of



(a) Number of constructed Trellis.



(b) Depth of the Trellis-based VRS.

Fig. 5. VRS Construction characteristics for different topologies.

our proposal. We tested the heuristic we propose by using a trellis of size 2^3. This parameter remains constant for all of the following simulation results. To evaluate simulation results, we computed the theoretical minimum number of trellis and the minimum depth of the recursive structure one can obtain for a given number of nodes.

5.1 Number of trellises

Figure 5(a) shows the number of trellis graphs produced on the two kinds of topology and the theoretical minimum. On a topology where all nodes are able to communicate in 1 hop (on a clique), our heuristic produces the minimum possible number of trellises. This indicates that the heuristic behaves correctly. This topology is nevertheless an ideal case, as the more nodes are connected, the easier it will be to fill each trellis with new nodes and thus reduce the number of new trellises. For more realistic topologies, where nodes are placed randomly, the upper curve shows that our heuristic remains close to the optimum number of trellises.

5.2 Depth of the VRS

Figure 5(b) shows that the depth of the structure constructed on such a fully-connected topology is very high. This is explained due to the algorithm of our heuristic. Since there is no physical topology constraint in this situation, the first trellis tested for an available position will accept the new node. The other possible trellises are never tested and this leads to a very high depth. We can also see that the curve representing the same situation increases but stays reasonably close. It shows that in a more realistic situation our heuristic behaves pretty well, even with more physical connectivity constraints. Optimizing both the number of trellises and the depth of the structure is really difficult, and can be the next improvement to this heuristic.

6 Conclusions

In this paper we have proposed the use of virtual Trellis structures to (a) organize the addressing space of a SON in a structured fashion, and, (b) to perform data routing in such an environment. Trellis structures are well known in classical communication and information theory, but their application in SON is novel. We strongly believe that introducing virtual structures for addressing space that are not based on trees will open new research areas. Thanks to the redundancy and recursion introduced in the trellis-based VRS, we provide a distributed dynamic addressing scheme, with the following advantages: localized operations, no size limit in the addressing space, a built-in multi-path routing structure, computed locally, and robustness to various kinds of mobility. The limited size of the trellis and the definition of local operations allow the addressing and routing to be highly scalable. The main drawbacks of our approach are (a) routes are suboptimal, in terms of number of hops, and, (b) control overhead is not fairly distributed among nodes.

References

1. R. Wakikawa, J.T. Malinen, C.E. Perkins, A. Nilsson and A.J. Tuominen: Global Connectivity for IPv6 Mobile Ad Hoc Networks. Internet-draft, IETF (2002)
2. D.B. Johnson and D.A. Maltz: Dynamic Source Routing in Ad Hoc Wireless Networks. Mobile Computing, Kluwer Academic Publishers **353** (1996)
3. S. Nesargi and R. Prakash: MANETConf: Configuration of Hosts in a Mobile Ad Hoc Network. In: Proc. of IEEE INFOCOM. (2002)
4. H. Zhou, L.M. Ni and M.W. Mutka: Prophet address allocation for large scale MANETs. In: Proc. of IEEE INFOCOM. (2003)
5. C.E. Perkins, E.M. Royer and S. Das: Ad Hoc On Demand Distance Vector (AODV) Routing. RFC 3561, IETF (2003)
6. C. Adjih, T. Clausen, P. Jacquet, A. Laouiti, P. Minet, P. Muhlethaler, A. Qayyum and L. Viennot: Optimized Link StateRouting Protocol. RFC 3626, IETF (2003)
7. M. Mauve, J. Widmer and H. Hartenstein: A Survey on Position-based Routing in Mobile Ad Hoc Networks. IEEE Network Magazine **6** (2001) 30–39
8. 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. (2000)
9. I.Stoica,R.Morris,D.Karger,M.F.Kaashoek,H.Balakrishnan: Chord:A Scalable P2P Lookup Service for Internet Applications. In: Proc. of ACM SIGCOMM. (2001)
10. A.C. Viana, M.D. Amorim, S. Fdida and J. F. Rezende: An Underlay Strategy for Indirect Routing. ACM Wireless Networks **10** (2004) 747–758
11. M.Castro, M.Costa and A.Rowstron: Should we build Gnutella on a structured overlay? In: HotNets-II. Cambridge, MA, USA. (2003)
12. S.D.Nikolopoulos, A.Pitsillides, and D.Tipper: Addressing Network Survivability issues by finding the K-best paths through a Trellis Graph. In: Proc. of IEEE INFOCOM. (1997)
13. S.Lin, D.J.Costello: Error Control Coding: Fundamental and Applications. Electrical Engineering Series. Prentice-Hall (1983)
14. J.Ridoux, A.Fladenmuller,K.Salamatian and Y.Viniotis: Beyond the Tree Structure: a new way to configure nodes in SONs. Technical Report, UPMC-LIP6 (2004) http://www-rp.lip6.fr/∼ridoux/Publications/TR_BeyondTrellis.pdf.