Implementation of Gradient Routing in Wireless Sensor Networks

Thomas Watteyne*, Kris Pister*, Dominique Barthel†, Mischa Dohler‡, Isabelle Auge-Blum§
*BSAC, UC Berkeley, USA {watteyne,pister}@eecs.berkeley.edu
†Orange Labs, Meylan, France, dominique.barthel@orange-ftgroup.com
‡CTTC, Castelldefels, Barcelona, Spain, mischa.dohler@cttc.es
§INRIA A4RES, CITI Lab, Lyon, France, isabelle.auge-blum@insa-lyon.fr

***Abstract*—IETF ROLL has recently proposed gradient routing as a fundamental building block for routing in Wireless Sensor Networks. This paper seconds this choice by presenting an implementation of gradient routing on current hardware, and by showing experimentally that gradient routing is robust against topological changes.**

To stress its self-healing quality, we design and implement a complete communication stack in which neighbor tables are built in a purely reactive fashion. We quantify the resulting topological changes, and show how gradient routing elegantly handles these dynamics.

This paper presents, to the best of our knowledge, the first experimental study on gradient routing as advocated by IETF ROLL.

I. Introduction

In a multihop network, a gradient routing protocol assigns a scalar value to each node, which we call its *height*. Heights are assigned in such a way that they increase with distance to a central node. Distance is calculated using a cumulative cost function which can be based on hop count, energy consumption, residual node energy, or any combination thereof. The forwarding process selects the next hop as the neighbor which offers the largest gradient, i.e. the neighbor with lowest height.

The concept of gradient is particularly useful for convergecast networks such as Wireless Sensor Networks (WSNs). In the simplest convergecast network, all traffic is sent to a single sink node. In this case, a single gradient – rooted at the sink node – is built and maintained in the network. Fig. 1 depicts a topology where nodes are assigned heights calculated as a function of hop count. When node Y at height 3 sends a message, it sends it to its neighbor of smallest height I; similarly I relays the message to G, and G to A.

The goal of the IETF work-group ROLL is to standardize a routing protocol for Wireless Sensor Networks. It recently selected gradient routing as a fundamental building-block of routing protocols for Wireless Sensor Network [1]. The goal of this paper is to confirm experimentally the proposal done by IETF ROLL: to show that gradient routing can easily be implemented in constrained WSNs and that it is robust against topological changes. Furthermore, we stress the importance of neighbor discovery at the MAC layer. This paper is to our knowledge the first paper to implement the gradient-based solution as considered by the IETF.

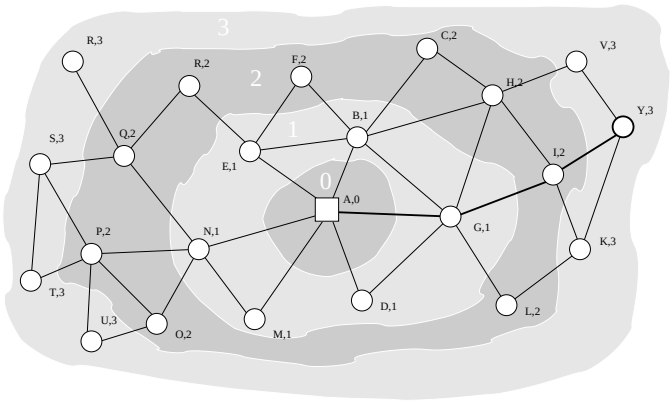


Fig. 1. Illustrating gradient routing. Nodes are attached *[Id,Height]*.

The remainder of this paper is organized as follows. Section II gives an overview of the gradient routing protocol proposed by the academic community, as well as the recent standardization efforts made in that direction. Section III presents the implemented protocol stack, where a reactive neighbor-discovery protocol is used to support gradient routing; implementation details are given in Section IV. Section V analyzes the implementation complexity as well as the energy-efficiency and obtained graph stability of the solution. Section VI concludes that gradient routing can easily be implemented in resource-constrained WSNs – hence confirming the choice of IETF ROLL – before presenting research directions.

II. Related Work

A. Design Driver Taxonomy

In this section, we present a comprehensive overview of proposed gradient routing protocols by classifying them according to their design driver.

a) Gradient setup: Gradient Based Routing (GBR) [2] sets up a hop-count-based gradient during a (possible repeated) setup phase. In the setup phase, the sink node issues a message containing a counter set to 1. When receiving this message, a node sets its height to the counter in the message, increments this counter by one, and relays the message to its neighbors. Using timers, such a gradient setup can be performed by having each node send only one packet [3].

b) Height calculation: The height of a node can be modulated by other factors than hop count to the sink. In [2] a node with low battery increases its height so that messages flow around it, relayed by nodes with more energy. Liu *et al.* [4] use a function of the node's neighborhood to modulate its height. This results in lesser nodes having the same height and a smoother gradient.

c) Gradient maintenance: Because nodes and link can appear/disappear in the network, a gradient needs to be maintained. Most proposed solutions [2], [3] rely on an independent setup phase. Whereas this phase can be rerun periodically, it is not clear what that period should be, and what to do with data packets during that gradient rebuilding phases.

Alternatively, [5] proposes to piggyback gradient setup messages in `Hello` messages. These messages are periodically exchanged between neighbor nodes to maintain neighbor tables; [5] proposes to add a height field into those packet. Although this removes the need for a periodic setup phase, `Hello` packets induce significant overhead at low network load.

[6] proposes to piggyback gradient setup information in data packets. This solution is energy efficient when combined with a MAC protocol which maintains neighbor table on-demand.

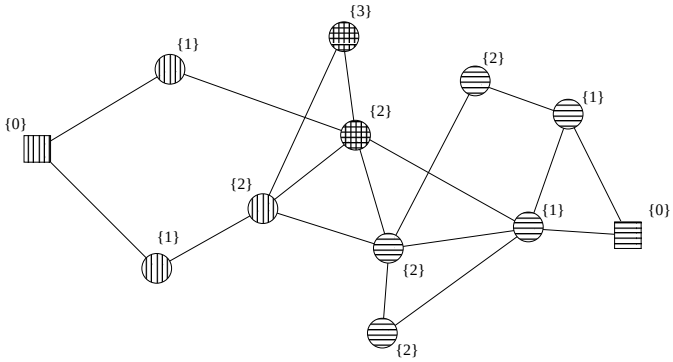
d) Forwarding techniques: GRAdient Broadcast (**GRAB** [7]) assumes a pre-established gradient. GRAB adds a credit field to the packet header; a relaying node decrements this credit by the energy to communicate this packet from its upstream neighbor, and sends it *to all its neighbors* if the remaining credit is higher than zero. The higher the initial credit field, the more nodes will simultaneously relay the same message. This, in turn, increases the overall delivery ratio of a message.

e) Multiple Sinks: [6] assumes the network contains a small number of sink nodes. When such a setting is used for robustness only – i.e. all sinks are equivalent – a single gradient is constructed, rooted at all sink nodes. A node implicitly sends a message to the sink which is closest (Fig. 2(a)). When the different sink nodes are not equivalent, multiple gradients are then built, one for each sink node; a node acquires multiple heights. To send a message to a particular sink, the node indicates which gradient to use in that packet (Fig. 2(b)).

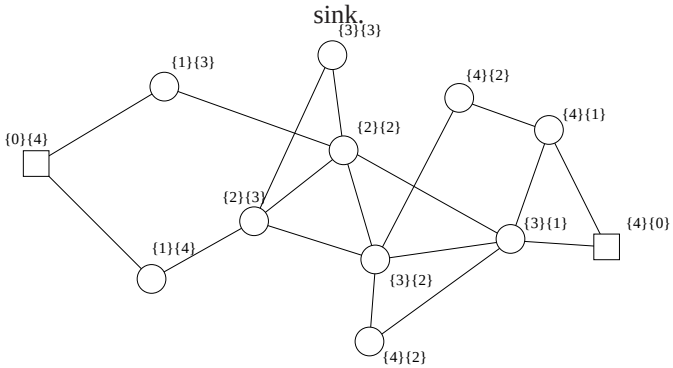
f) Gradient and geographic routing: Some geographic routing protocols use the set of heights $\{V_1, V_2, \ldots, V_N\}$ obtained from a set of N anchor nodes as a relative coordinates. Translating these relative coordinates of two nodes $V = V_1, V_2, \ldots, V_N$ and $W = W_1, W_2, \ldots, W_N$ into distance $||D_p||$ can be done using Eq (1) with $p = 2$ (as proposed in [8]) or $p = 10$ (as proposed in [5]).

$$||D_p|| = \sqrt[p]{\sum_{i=1}^{N} (V_i - W_i)^p}. \quad (1)$$

[8] and [9] show by simulation and experimentally, resp., that geographical routing over these gradient-based relative coordinates yields higher delivery ratios than using real geographical coordinates.



(a) Single gradient: data is sent to the topologically closest sink.



(b) Multiple gradients: nodes can choose the destination sink.

Fig. 2. Using gradients with multiple sinks.

B. Standardization Efforts

The IETF Routing Over Low power and Lossy networks (ROLL) working group is designing a routing protocol for WSNs. It has recently advocated gradient routing as one of its fundamental building blocks [1]. IETF ROLL proposes to use a `TreeDepth` parameter to the IPv6 Neighbor Discovery Router Advertisements, which are sent periodically by each node.

C. Motivation and Goal

Acknowledging the importance of gradients in routing protocols for WSN in academic and standardization bodies, the goal of this paper is to experimentally confirm the choice by IETF ROLL. We present a complete protocol stack in which the MAC layer does not hide physical topological dynamics in order to demonstrate the gradient routing's inherent self-healing characteristics. A secondary goal is to present the production-quality implementation and associated design challenges. This is, to our knowledge, the first paper to implement gradient routing as advocated by IETF ROLL.

III. Implemented Protocol Stack

We present the functions of the routing and MAC layers in Sections III-A and III-B, resp., before presenting a comprehensive overview in Section III-C.

A. Hop-Count-Based Gradient Routing

This routing concept has been chosen by IETF ROLL for its simplicity and validation through extensive use in routing protocols. Our goal is to show that this concept can be used efficiently in WSNs. Borrowing the design-driver taxonomy from Section II, the gradient routing protocol implemented has the following characteristics:

- Gradient setup information is piggybacked in data messages. The novelty of this approach lies in the fact that there is no periodic signaling traffic, which enables for ultra-low power operation when the network sits idle.
- Height calculation: When a node switches on, it sets its `Height` to NaN (Not a Number, a non-scalar value), except the sink node which sets it to 0. Whenever is transmits or relays a message, it learns its list of neighbors (through a MAC mechanism, see next section) and updates its `Height` by the minimum height of its neighbors', incremented by one. If all its neighbors have a height set to NaN, it sets is `Height` to NaN. The sink node always keeps its `Height` at 0. For simplicity, the gradient which is built is hop-count-based, although any other metric can be easily used.
- Forwarding techniques: A node forwards a packet to its available neighbor with smallest `Height`. The originality is that a node does not have a single routing parent, but rather forwards to the best available node, hence increasing network reliability through multiple routing paths.
- Multiple Sinks: for simplicity, a single gradient is formed into the network as all sinks are assumed to be equivalent. Using multiple gradients does not affect the results.

B. Reactive Neighbor Discovery

Neighbor Discovery is a essential component of the MAC layer; we have chosen to use a simplified version of the 1hop-MAC protocol [10]. The originality of this protocol is that it rebuilds the neighbor list each time a messages is sent. Because state is not maintained and neighbors and not enforced through link hysteresis (i.e. a link failure is immediately reflected in the neighbor table), radio link dynamics are reflected at the routing layer. 1hopMAC is used to show how the routing is able to withstand these dynamics; this can be seen as worst case scenario as using a state-full MAC protocol would result in less topological changes.

The MAC protocol functions as follows. When a node sends a message, it broadcasts a request to its neighbors and opens a window for them to announce their presence. As announcements are received, the node builds a neighbor table. After the packet has been sent, the neighbor table is discarded; it will be rebuilt at the next transmission.

For energy efficiency, nodes use preamble sampling when idling, i.e. a node listens for a very short interval D_{check} every check interval Check Interval (CI). This behavior is detailed in the next section.

C. Overview and Protocol Timeline

In the implemented protocol stack, the application layer generates sensed data to be sent to a sink node, by using on-board Analog-to-Digital Conversion. The routing layer is responsible for updating the node's `myHeight`; the MAC layer performs on-demand neighbor discovery and uses preamble sampling for energy-efficiency.

The execution timeline of the implemented protocol is presented in Fig. 3 for an example topology of 3 nodes. By default, nodes perform preamble sampling. When a node wants to send a message (here A), it starts by sending a preamble as long as the check interval (CI) to make sure all neighbors hear that preamble. For efficient handling by a packet radio, the preamble is cut into a series of micro-frames UF, each containing a counter indicating the number of UF still to come. Upon hearing a UF, a receiving node turns its radio off and sets a timer to switch into receive mode after the last UF. At that moment, the sender indicates the duration of the neighbor announcement window to follow in a CW packet.

Receivers choose a random backoff for sending an ACK message inside the neighbor announcement window and sleep the rest of the time; the sender listens for the complete announcement window and populates the initially empty neighbor table as it receives ACK messages.

After the neighbor announcement window, the sender updates its `myHeight` by the minimum value of its neighbors', incremented by one, and select its neighbor with smallest `Height`. It inserts this information into the DATA packet header which it transmits. The destination node receives the whole packet while the non-destination neighbor switches to sleep after the header. The destination replies with a final acknowledgment FIN; all nodes resume preamble sampling.

IV. Implementation Details

A. Development Environment and Hardware

We implement this protocol on the popular TI eZ430-RF2500 platform (MSP430 microcontroller with 32kB ROM/1kB RAM and 2.4 GHz CC2500 radio) because of its very reduced form factor and price, and because its radio can perform preamble sampling as a standalone device. We evaluate the advantages of the latter in Section IV-C.

Implementation is done on the bare hardware without Operating System to allow for fine-grained optimization and small memory footprint. Development is done in C using the IAR toolchain[1]. Parameters used for the experiment are listed in Table II.

B. Packet Formats and Parameters

We list the packet formats in Table I. `Type` indicates the type of frame. The DATA frame contains a sequence number `seq` incremented by the sender at each new DATA frame, the internal temperature of the MSP403 (`temperature`), the value of the 10 ADC (`ADC1` ... `ADC10`) and the battery voltage (`battery`).

[1]The fully IAR C source code is freely available upon request.

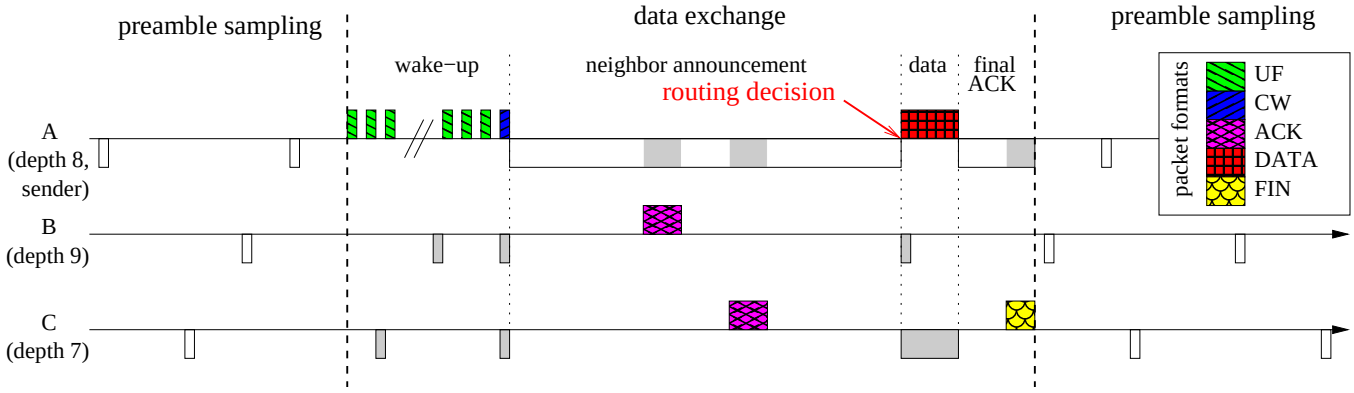


Fig. 3. Timeline illustrating the execution of the protocol stack. The x-axis represents time; a box above the line indicates that the radio is transmitting; a gray/white box under the axes means that the radio in receiving/idle listening, resp.; no box means the radio is turned off.

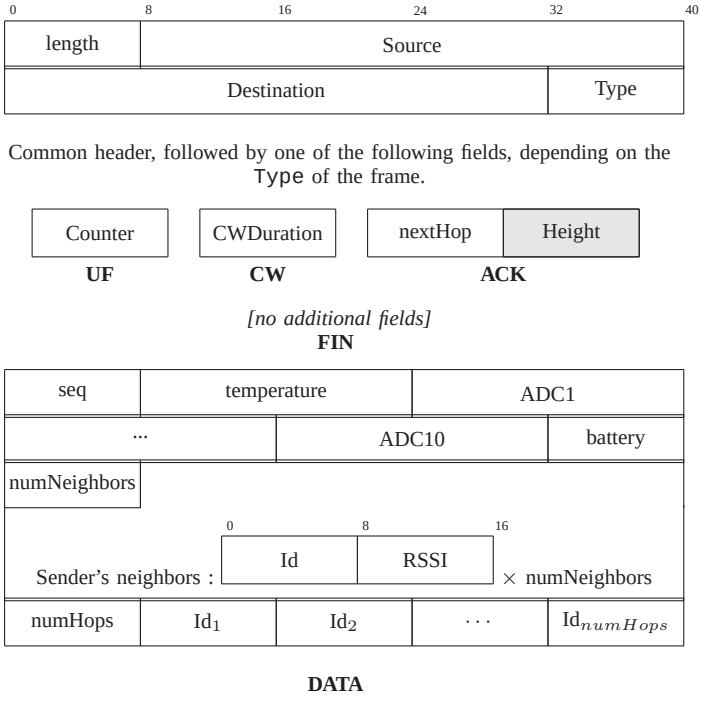


TABLE I
FORMAT OF THE FIVE FRAME TYPES. THE SHADED FIELD INDICATES THE ONLY COMMUNICATION OVERHEAD REQUIRED BY GRADIENT ROUTING.

For debugging purposes, the DATA frame also contains the list of neighbors of the sending node, including the degree[2] (`numNeighbors`), the identifier of each (`Id`) and the receive signal strength indicator when receiving their ACK (`RSSI`). For debugging the DATA frame additionally contains the length of the multi-hop path followed by the packet before reaching the sink node (`numHops`), as well as the identifiers of the traversed nodes ($\texttt{Id}_1 \ldots \texttt{Id}_{numHops}$).

[2]The degree of a node is defined as its number of neighbors.

C. HW vs. SW Preamble-Sampling

The CC2500 radio chip supports preamble sampling in hardware. The goal of this section is to decide whether this hardware (HW) support yields significant energy savings over a purely software (SW) implementation in which the microcontroller drives the radio to perform preamble sampling. We implement both solutions and measure the energy consumed by the board in both cases. Note that gradient routing present by IETF ROLL does not make any assumption on the MAC protocol; preamble sampling can easily be replaced.

g) HW solution: This is available in radio chip such as Texas Instruments' CC1100, CC1101 and CC2500. The microcontroller configures the chip to start the preamble-sampling sequence, and is then put to sleep. If the start of a message is detected, the radio chip interrupts the microcontroller, which can then take the appropriate action. HW support enables simple code.

h) SW solution: The microcontroller handles one timeout for measuring CI, another for D_{check}. Our test implementation sources these two timeouts from two different clocks available in the microcontroller for energy-efficiency.

i) Energy Consumption Measurements and Comparison: We measure the current consumption of a channel sample by reading the voltage off a 1Ω resistor mounted on the power supply of the board, for both the HW/SW solutions. Table III details the different phases during a channel check. On average, the SW solution consumes around 2% more energy than HW solutions. The implementation presented in this paper uses the CC2500 to perform preamble sampling; this section shows that an alternate SW solution would consume only 2% more.

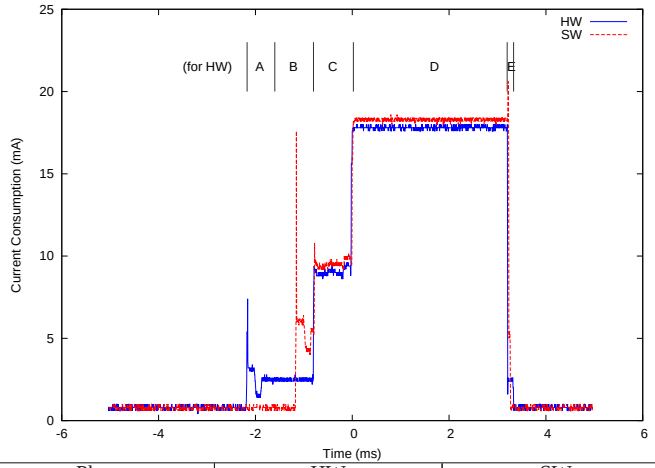
V. IMPLEMENTATION RESULTS

A. Code Size and Complexity

MAC and routing layers have been implemented in 571 lines of C-code, with 200 lines for core MAC operations and only 30 lines for routing. The memory footprint of the implementation is 7440 bytes of code memory an 850 bytes of data memory.

parameter	value	comment
message generation		
transmission period	5sec+rand(10sec)	removing and transmitting a packet from the Tx queue
data packet generation	3	inserting a packet into Tx queue
preamble sampling		
CI	104ms	
D_{check}	3.24ms	time to perform a radio check
number of UF	53	(CI/UF period)+3 for timing security
MAC timing		
UF period	2ms	delay between two consecutive UF in preamble
MAC abort watchdog timer	60ms	when not receiving an expected packet
MAC guard time	5ms	early wakeup to take clock drift into account
Tx queue management		
size	3 packets	
retries	3 times	a packet is dropped when number of retries is reached
PHY configuration		
Tx power	-14dBm	indoor communication range of 3-5m

TABLE II
PARAMETERS USED DURING IMPLEMENTATION.



Phase		HW		SW	
		duration	av. current	duration	av. current
idle		98.6ms	0.800mA	99.6ms	0.788mA
A	μC startup	0.512ms	2.60mA	0.396ms	5.45mA
B	radio startup	0.800ms	2.48mA	0	NA
C	radio freq. cal.	0.824ms	9.03mA	0.772ms	9.55mA
D	reception mode	3.176ms	17.7mA	3.24ms	18.3mA
E	entering sleep	0.132ms	2.68mA	0.032ms	5.00mA
average consumption		1.427mA		1.450mA	

TABLE III
CURRENT CONSUMPTION OF THE HW AND SW SOLUTIONS AT THE RECEIVER (THIS LISTEN PERIOD REPEATS EVERY CI). RESULTS AVERAGED OVER 128 SAMPLES.

Gradient routing is simple and can be easily implemented on current WSN hardware. This is mainly because gradient routing does not maintain routing state other than the 8-bit `myHeight` variable. Scalability is hence ensured in O(1), a characteristic required for WSNs by IETF ROLL, as stated in [11].

B. Energy Efficiency

Energy efficiency and lifetime calculations depend largely on topology and network load. The different elements which consume energy are:

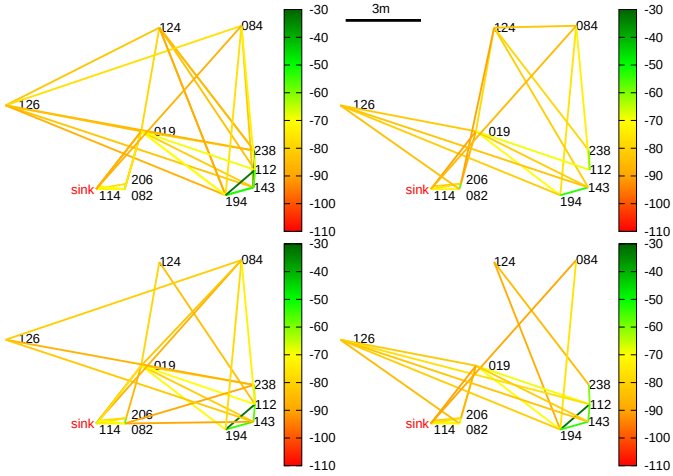
- The communication phase, which lasts for 176ms (108ms for the UF and CW packets, 60ms for the neighbor announcement window, 8ms for DATA and FIN packets). During a communication phase, the sender consumes on average 12mA, the receiver consumes on average 8mA.
- During preamble sampling, a single channel sample costs 67.3μJ (see Table III). The average consumption when the network sits idle depends on the check interval CI, e.g. 1.427mA for $CI = 104ms$. This can be brought down by increasing CI, at the expense of longer preambles. The optimal values depends on the network load.

In a typical urban application where a node running on two standard AA batteries (approx. 4000mAh), has on average 5 neighbors and sends on average 5 messages per hour at 1dBm, with a $CI = 500ms$, it would run for 4 months. This value can be increased by carefully tuning the value of CI, depending on the network load.

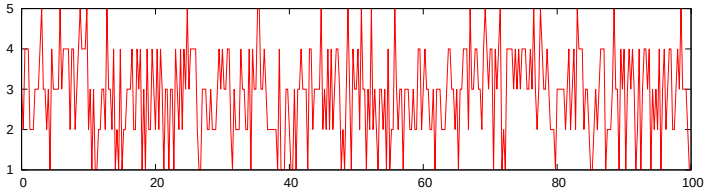
C. Routing Graph Dynamics and Delivery Ratio

A routing protocol should cope with the network dynamics inherent to WSNs. We therefore stress the self-healing nature of IETF ROLL's gradient routing by deploying 12 nodes during 8 hours, sufficiently far from each other to obtain weak links. Link outages trigger continuous topological changes (see Fig. 4) which are not hidden by the MAC layer and thus need to be coped with at the routing layer.

Table IV quantifies the node degree variation and shows that, despite the constantly changing topology, three out of four sent packets reach the sink node. In such highly dynamic environments, flooding based protocols fail as reverse links are out dated; similarly, topological changes generate reorganization traffic in clustered solution which saturate the network, provoking network collapse.



Snapshots of the topology at four different times, with nodes at their geographical location. The color of the segment indicates the RSSI in dBm of the links between neighbors.



Degree of node 112 evolving over 100 minutes.

Fig. 4. Witnessing topology dynamics.

Id	Hop Count		Degree		PDR*
	Average	Std.Dev.	Average	Std.Dev.	
206	1.06	0.33	3.34	1.21	1.00
238	2.53	0.93	3.75	1.44	0.42
082	1.12	0.41	4.07	1.44	1.00
124	2.65	0.80	2.65	1.30	0.44
114	1.02	0.15	3.02	1.04	1.00
143	2.51	0.85	3.76	1.32	0.39
019	1.90	0.58	5.34	1.81	0.88
126	2.55	0.82	2.30	1.29	0.39
084	1.15	0.60	4.45	1.56	0.95
194	3.18	0.88	2.97	1.26	0.45
112	3.22	1.03	3.08	1.29	0.55
Cumulative	**1.91**	**0.61**	**3.49**	**1.33**	**0.74**

* Packet Delivery Ratio

TABLE IV
EXPERIMENTAL RESULTS OVER 6984 PACKET.

VI. CONCLUSIONS AND FUTURE WORK

This paper has shown that the recommendation made by IETF ROLL to use gradient routing in WSNs is valid in that (1) gradient routing can be easily implemented on WSNs hardware with negligible impact on memory footprint, complexity and signaling overhead, and (2) gradient routing is robust against topological changes and is inherently self-healing. Moreover, this paper presents a simple and complete communication stack which supports gradient routing. This paper contains, to the best of our knowledge, the first implementation results of gradient routing as advocated by IETF ROLL.

Gradient routing only allows for upstream data, flowing towards the sink node. Although most WSNs application are convergecast, downstream routing is necessary for controlling actuators, reconfiguration etc. Downstream multi-hop paths may be set up by maintaining forwarding tables at intermediate nodes as data flows upstream. We are currently working on designing such a protocol without losing the simplicity of gradient routing, by piggybacking information into the data packets.

Enhancing the protocol stack necessarily includes enhancing the MAC protocol, which has been kept purely reactive on purpose to stress the self-healing component of gradient routing. An improved MAC protocol ranks neighbors according to link quality, combined with a notion of confidence to avoid over reacting on transitory outages. We are currently working on combining such compound neighbor-quality metrics with metrics used for routing to improve the selection of the next hop node in order to meet lifetime of quality of service constraints.

REFERENCES

[1] P. Thubert, T. Watteyne, Z. Shelby, and D. Barthel, "LLN Routing Fundamentals," IETF ROLL, IETF Internet-Draft, 31 March 2009, draft-thubert-roll-fundamentals-00 [work in progress].

[2] J. Faruque, K. Psounis, and A. Helmy, "Analysis of Gradient-based Routing Protocols in Sensor Networks," in *International Conference on Distributed Computing in Sensor Systems (DCOSS)*. Marina del Rey, CA, USA: IEEE/ACM, June 2005, pp. 258–275.

[3] O. Powell, A. Jarry, P. Leone, and J. Rolim, "Gradient Based Routing in Wireless Sensor Networks: a Mixed Strategy," arXiv.org automated e-print archives, Report CS-0511083, Tech. Rep., 2005. [Online]. Available: http://arxiv.org/abs/cs/0511083

[4] K. Liu and N. Abu-Ghazaleh, "Aligned Virtual Coordinates for Greedy Routing in WSNs," in *International Conference on Mobile Adhoc and Sensor Systems (MASS)*. Vancouver, Canada: IEEE, October 9-12 2006, pp. 377–386.

[5] Y. Zhao, Y. Chen, B. Li, and Q. Zhang, "Hop ID: A Virtual Coordinate based Routing for Sparse Mobile Ad Hoc Networks," *IEEE Transaction on Mobile Computing*, vol. 6, no. 9, pp. 1075–1089, September 2007.

[6] T. Watteyne, I. Auge-Blum, M. Dohler, S. Ubda, and D. Barthel, "Centroid Virtual Coordinates - A Novel Near-Shortest Path Routing Paradigm," *International Journal of Computer and Telecommunications Networking, Elsevier, Special Issue on Autonomic and Self-Organising Systems*, vol. to appear., p. to appear., 2009.

[7] F. Ye, G. Zhong, S. Lu, and L. Zhang, "GRAdient Broadcast: A Robust Data Delivery Protocol for Large Scale Sensor Networks," *ACM Wireless Networks*, vol. 11, pp. 285–298, 2005.

[8] Q. Cao and T. Abdelzaher, "A Scalable Logical Coordinates Framework for Routing in Wireless Sensor Networks," in *25th IEEE International Real-Time Systems Symposium (RTSS)*. Lisbon, Portugal: IEEE, 5-8 December 2004, pp. 349–358.

[9] R. Fonseca, S. Ratnasamyy, J. Zhao, C. T. Ee, D. Culler, S. Shenker, and I. Stoica, "Beacon Vector Routing: Scalable Point-to-Point Routing in Wireless Sensornets," in *2nd Symposium on Networked Systems Design & Implementation (NSDI)*. Boston, MA, USA: ACM, 2-5 May 2005, pp. 329–342.

[10] T. Watteyne, A. Bachir, M. Dohler, D. Barthel, and I. Auge-Blum, "1-hopMAC: An Energy-Efficient MAC Protocol for Avoiding 1-hop Neighborhood Knowledge," in *International Workshop on Wireless Ad-hoc and Sensor Networks (IWWAN)*. New York, NY, USA: IEEE, June 2006, pp. 639–644.

[11] P. Levis, A. Tavakoli, and S. Dawson-Haggerty, "Overview of Existing Routing Protocols for Low Power and Lossy Networks," IETF ROLL workgroup, IETF draft, 14 February 2009, draft-ietf-roll-protocols-survey-06 (work in progress).