

Sense & Sensitivity: A Large-Scale Experimental Study of Reactive Gradient Routing

T. Watteyne¹, D. Barthel², M. Dohler³ and I. Augé-Blum⁴

¹BSAC, UC Berkeley, USA.

²Orange Labs, Meylan, France.

³CTTC, Castelldefels, Barcelona, Spain.

⁴Université de Lyon, INRIA, INSA-Lyon, CITI, F-69621, Lyon, France.

E-mail: watteyne@eecs.berkeley.edu

Abstract. This paper presents and evaluates a cross-layered communication architecture which allows for robust, reliable, and efficient data collection in an embedded wireless multi-hop network. The proposed solution, based on a novel embodiment of gradient routing, proposes to piggyback control data along with data packets. We show how this simple scheme is robust against topological changes which are quickly absorbed by the network, and that it is largely unaffected by the density or size of the network. Experimental results confirm these observations and show how the nodes and sinks can be added/removed without disturbing network operation while staying energy-efficient. The simple solution proposed in this paper offers a true deploy-and-forget user experience.

Keywords: Gradient Routing, Robustness, Experimental Study.

AMS classification scheme numbers: 94C99

Submitted to: *Meas. Sci. Technol.*

1. 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 from 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 that 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 *D* at height 2 sends a message, it sends it to its neighbor of smallest height (*B*); similarly *B* relays the message to *A*.

A wireless multihop network is a dynamic environment, even with static nodes. The heights of the nodes need to be updated as links come and go because of interference, multi-path fading, nodes being inserted/removed, or people moving inside the deployment area. This paper shows that a simple reactive approach is effective and efficient. For each transmitted packet, a node discovers its neighbors and updates its height. Simulation results show that this approach yields a stable gradient; large-scale experimental results confirm that it is energy-efficient.

The remainder of this paper is organized as follows. Section 2 shows that gradient-based routing is particularly interesting for WSNs and how it is pushed forward by standardization bodies such as the IETF. Section 3 describes the simple reactive update mechanism used for heights. Unlike other proposals, it does *not* involve rebuilding the graph through periodic broadcasts. It also shows by simulation that heights quickly adapt to topological changes, and that the reactive gradient routing protocol is appropriate for dynamic environments. Section 4 presents the results obtained from a large-scale experiment called Sense&Sensitivity. It shows how propagation characteristics cause the connectivity graph to change dynamically, and how the proposed reactive gradient setup elegantly copes with these changes. Section 5 concludes this paper by discussing the opportunities and limitations offered by reactive gradient routing.

2. Gradient Routing in WSNs

Routing in wireless multi-hop networks has received significant attention, and several families of solutions have emerged. Most of them apply to the more general family of Mobile Ad-hoc NETwork (MANET), motivated by the eponymous IETF Working Group[†]. These include the ever-popular AODV [1] and DSR [2], which use flooding to issue a request. The reply then follows the reverse path back to the requester. These protocols enable any node in the network to send data to any other node. When dealing with WSNs, the traffic pattern changes from any-to-any to any-to-some: a large group of nodes sends data to a small group of (possibly equivalent) sink nodes. The convergecast nature of WSNs makes routing somewhat simpler.

In the canonical case where there is only one sink, Gradient Based Routing (**GBR**) [3, 4] can be used. During a setup phase, the sink node issues a message containing a counter set to 1. When receiving this message, a node sets an internal variable called *height* to the counter in the message, increments this counter by one, and relays the message to its neighbors. This sets up a network-wide permanent gradient. Once the gradient is set up, routing can start. Each node keeps a neighbor table containing the height of its neighbors and sends a message to its neighbor with smallest height. A message therefore follows the gradient towards the sink much like an impatient hiker would follow the steepest slope towards the bottom of a valley. GBR is well-tailored to the needs of WSNs. As long as the connectivity graph does not change (i.e. links do not (dis)appear), the gradient ensures messages reach the sink in the minimum number of hops. Its height tells a node how many hops separate it from the sink.

GRADient Broadcast (**GRAB**) [5] enhances the reliability of data delivery through path diversity. GRAB builds and maintains a gradient, providing each sensor the direction to forward sensing data. However, unlike all the previous approaches, GRAB forwards data along a band of interleaved mesh from each source to the receiver.

[†] <http://www.ietf.org/html.charters/manet-charter.html>

To collect data reports, the sink first builds a gradient by propagating advertisement (ADV) packets in the network. The height at a node (dubbed “cost” in GRAB) is the minimum energy overhead to forward a packet from this node to the sink along a path; nodes closer to the sink have a smaller cost. GRAB makes the assumption that each node has the means to estimate the cost of sending data to nearby nodes, e.g., through SNR measurements of neighbors’ transmissions. Each node keeps the cost of forwarding packets from itself to the sink. Since only receivers with smaller costs may forward the packet at each hop, the packet is forwarded by successive nodes to follow the decreasing cost direction to reach the bottom of the cost field, which is the sink.

Multiple paths of decreasing cost can exist and interleave to form a forwarding mesh. To limit the width of this mesh in order to avoid creating excessive redundancy and wasting resources, a source assigns a credit to its generated packet. The credit is some extra budget that can be consumed to forward the packet. The sum of the credit and the source’s cost is the total budget that can be used to send a packet to the sink along a path. A packet can take any path that requires a cost less than or equal to the total budget. Multiple nodes in the mesh make collective efforts to deliver data without dependency on any specific node.

Performance analysis of GRAB shows the advantage of interleaved mesh over multiple parallel paths and shows that GRAB can successfully deliver over 90% of packets with relatively low energy cost, even under the adverse conditions of node failures and link message losses.

In this case, a strip of nodes between source and sink nodes all relay the message; the larger the strip, the higher the delivery ratio (and the energy consumed). This is the idea behind GRADient Broadcast (**GRAB**) [5]. A gradient is set up using previously described techniques. During run-time, each message contains a credit field which is decremented at each hop; a message is discarded when the credit reaches 0. The message’s credit is decremented at each hop by the difference between the heights of the nodes on that link. The source node controls the width of the strip of relaying nodes by allocating more or less credit to the message it issues.

The height is not necessarily a function of hop count only. In [3], a node with low battery increases its height so that messages flow around it, relayed by nodes with more energy. Liu *et al.* [6] use a function of the node’s neighborhood to modulate its height. This results in fewer nodes having the same height and a smoother gradient.

A surprising twist can be applied to gradient to perform any-to-any routing. Multiple gradients can be built simultaneously in the network, each rooted at a different “anchor” node. A node then acquires a vector $V = \{V_1, V_2, \dots, V_N\}$ where V_i is the hop distance from the current node to anchor node i and N the number of anchor nodes. Applying Euclidean distance to these relative coordinates enables geographic routing protocols to find multi-hop paths from any node to any node. This technique comes with its own set of challenges, such as placing enough anchor nodes, achieving a high delivery ratio and stabilizing the relative coordinates [7–9].

Any-to-some traffic is more applicable to WSNs. Whereas multiple sinks may exist, they may be equivalent, and a data message can be sent to either one. Such a behavior comes naturally with gradients when all sink nodes have their height at $\{0\}$. As shown in Fig. 1, nodes send their messages to the sink topologically closest. Additional sink nodes can be introduced without requiring any configuration.

The **IETF** – through its Working Group **ROLL**[†] – has recently identified gradient routing as an important building block for the to-be-standardized WSN routing protocol called RPL [10]. In RPL, a gradient is built and maintained by using DAG Information Option (DIO), as a special type of ICMPv6 messages. DIO messages originate at the sink node and propagate outwards, setting up a gradient according to a metric specified in the DIO.

So far, we have considered periodic gradient reconstruction by sending control messages at a fixed interval. As stated in [11], this interval poses a basic tradeoff. A small interval reduces how stale information can become, but it uses more bandwidth and energy. A large interval uses less bandwidth and energy but can let topological problems persist for a long time. The Collection Tree Protocol (**CTP**) [11] therefore uses the Trickle algorithm [12] to regulate the beaconing interval. In the absence of topological changes, this interval is regularly doubled until it reaches a maximum preset value. Upon topological changes, the interval is reduced to allow for fast gradient reconvergence.

[†] <http://www.ietf.org/html.charters/roll-charter.html>

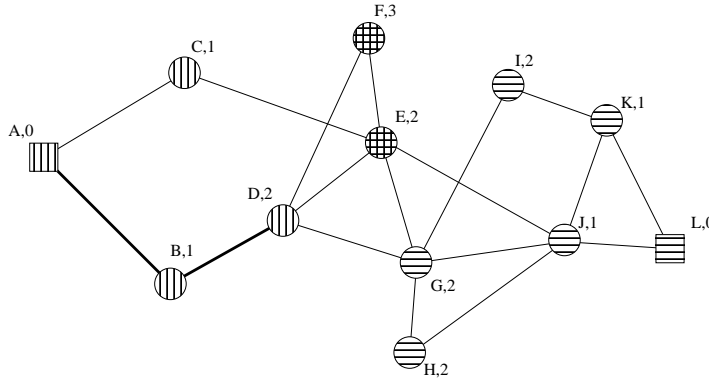


Figure 1. Multiple sinks – represented as squares – can be used simultaneously without requiring any configuration. Nodes – represented as circles – implicitly send data to the sink topologically closest. Numbers represent the nodes’ height. The fill pattern indicates to which sink a node sends its data. Some nodes may send their data to either of the sink nodes; these are represented with both fill patterns.

3. Reactive Gradient Routing

Because gradient routing is particularly suited for convergecast WSNs, it has been in use for almost a decade and is known as Gradient Based Routing [3, 4], Gradient Broadcast [5], Directed Acyclic Graph Routing [10], or Collection Tree Routing [11]. CTP has recently shown that this approach can yield high data delivery ratios. Moreover, CTP adapts the gradient refresh rate as a function of the topology dynamics. We go one step further by freeing the protocol from beaconing altogether. Instead of periodically sending control traffic on a network which may be idle, we propose to piggyback gradient information in the data packets. This way, the network consumes no energy due to gradient maintenance when it idles, but converges rapidly to an optimal state when data is sent. We call this approach Reactive Gradient Routing, the design-drivers for which are the following:

- **Reactivity.** The absence of an initialization phase enables the network to be operational immediately after deployment.
- **Robustness.** Heights are continuously updated throughout the lifetime of the network and quickly reflect topological changes due to nodes (dis)appearing, mobility, or link dynamics.
- **Energy-Efficiency.** The update process occurs only when a message is sent, which is especially beneficial when the network load is low or bursty. We couple the update process with a Medium Access Control (MAC) layer protocol to guarantee energy efficiency.
- **Multiple Sink Nodes.** In the presence of multiple sink nodes, a node sends its message to the topologically closest sink node. Sink nodes can be added, removed or relocated in the network without requiring reconfiguration.

3.1. Initialization and Gradient Bootstrapping

When switched on, a node sets its height to NULL, a non-value. Whenever a node intends to send a message, it scans through the heights of its neighbors by means of a suitable MAC protocol, finds the smallest one, and updates its virtual height to this value incremented by one. The routing protocol uses the field of heights to greedily route towards a sink node. A node thus needs to retrieve the heights of its neighbors as part of the MAC protocol, an embodiment of which is detailed in the next section.

Topological changes impact the connectivity graph, and can create local minima in the routing gradient. In non-reactive gradient routing, such inconsistencies can result in packets not being able to progress until the gradient is rebuilt. In the solution proposed in this paper, possible inconsistencies are corrected at every message transmission. Even when a node is in

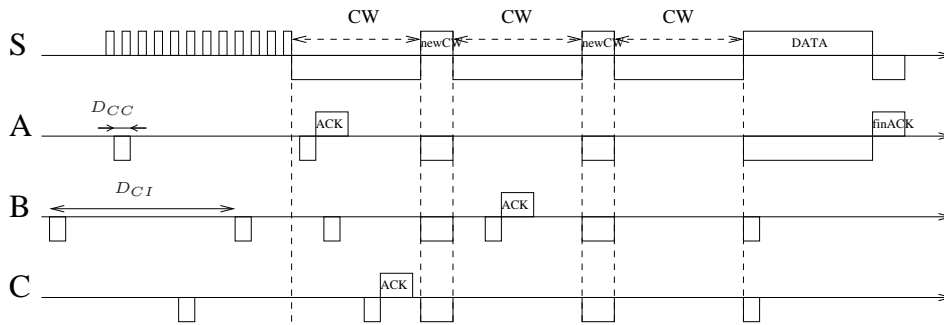


Figure 2. Chronograph representing the radio state of nodes S , A , B and C when executing 1-hopMAC. A box above/below a vertical line indicates the node’s radio is transmitting/receiving, respectively. No box means the radio is off.

a local minimum, when it has a packet to send, it updates its height to a value larger than at least one of its neighbors, removing itself from the minimum.

3.2. Medium Access Control Protocol

We use preamble sampling at the MAC layer for energy efficiency. Preamble Sampling (this technique has also been referred to in literature as Cycled Receiver, Low Power Listening or Channel Polling) is a power-saving technique used in MAC protocols for WSNs. Nodes using preamble sampling are not synchronized. Instead, nodes periodically listen for a very short time (called Check Duration, or D_{CD}) to decide whether a transmission is ongoing. We call check interval the amount of time a node waits between two successive CDs (D_{CI}). The sender needs to make sure the receiver node is awake before sending data; it thus prepends a *long* preamble to its data. By having the preamble at least as long as the Check Interval D_{CI} , the sender is certain the receiver will hear it and be awake for receiving the data.

We perform neighbor discovery between the preamble and the data. Fig. 2 depicts the timeline of the resulting MAC protocol for a network of 4 nodes, where node S sends a message. The resulting protocol – called 1-hopMAC – was previously published in [13]. After the preamble, all neighbors of S pick a random backoff time inside a contention window CW ; when that backoff timer elapses, a node listens to the medium and sends a message indicating its height if the medium is free. If the medium is occupied by another *ACK* message (this is the case for node B), the sender opens up a new window by issuing a *newCW* message. It will start new contention windows until it receives no further *ACK* packets. At that time, the node picks the neighbor with smallest height, updates its height, and sends the data packet. A *finACK* packet acknowledges the successful reception of the data at the next hop. Details of the protocol, including the exact fields contained in each of the frame formats, are given in [14].

3.3. Simulation Results

As a first step for understanding the behavior of a large-scale network running Reactive Gradient Routing, we use a home-made packet-level simulator[†] which features a simple Unit Disk Graph propagation model. At each simulation, 100 nodes are placed randomly in a 1000×1000 unit square area. Radio range is set to 200 units, which yields an average node degree of 12.5. This topology is different for each run, and we average the results over 10^5 runs. We calculate and depict a 95% confidence interval; the smaller that interval, the more “confidence” one can have in presented averaged results.

Messages are sent from randomly chosen connected nodes to the sink. Whereas we simulate the routing protocol by sending packets in the network one at a time, the results are strictly identical in the general case where multiple packets flow in the network simultaneously. This is because the resulting contention is dealt with at the medium access control layer,

[†] Source code available at <http://openwsn.berkeley.edu/>.

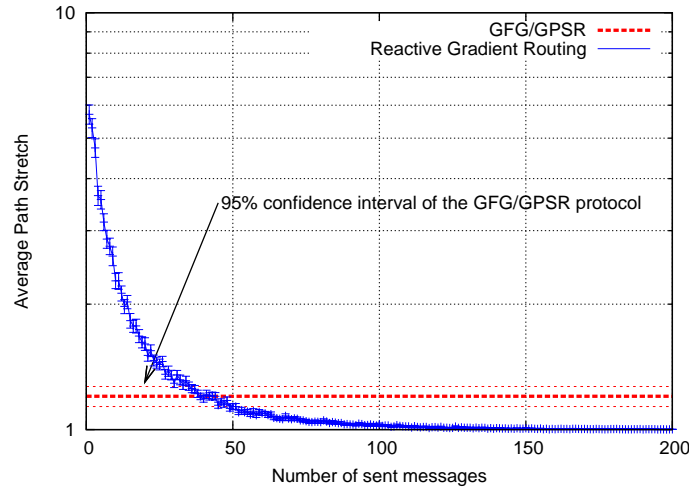


Figure 3. The average path stretch when using Reactive Gradient Routing, compared to the GFG/GPSR protocols with an average node degree of 11. Simulation results are presented with a 95% confidence interval.

without impacting routing decisions. We use the number of sent messages as a metric of time. Assuming each hop consumes the same amount of energy, we assimilate energy efficiency to hop count. Because of their wide-spread acceptance, we use Greedy Face Greedy (GFG)/Greedy Perimeter Stateless Routing(GPSR) [15] as a baseline comparison. For an unbiased comparison, simulation results are presented at high network density, where GFG/GPSR give good results.

Convergence. Fig. 3 is obtained in the extreme case where all nodes are switched on simultaneously, without knowing their height. It depicts how the path stretch (the ratio between the actual and minimal number of hops) evolves as packets flow through the network. Whereas the first packet wanders around the network for many hops, the following ones quickly find shorter paths and reach a path stretch very close to the *optimal* value of 1 after about 100 messages. If each node sends one packet every 10 seconds, this state would be reached 10 seconds after the nodes are switched on.

These results indicate that the gradient converges quickly after important topological changes. This self-healing characteristic is important in dynamic wireless environments. Since GFG/GPSR base their routing decision purely on geographic information, the network does not “converge”. The path stretch instead remains at 1.13, i.e. discovered paths are 13% longer (in number of hops) than the optimal case. Reactive gradient routing converges to a near-optimal case. It therefore outperforms GFG/GPSR, despite the fact that it requires an initial convergence phase.

The following paragraphs quantify the effect of average node degree and number of nodes on the convergence speed; the faster the converge, the lower the resulting average path stretch. GFG/GPSR are not analyzed in what follows as these protocols do not exhibit convergence.

Impact of average node degree. To evaluate the impact of average node degree, we set the communication range of each node to 200 units and choose the number of nodes as a function of the desired average node degree. 1000 runs are performed; for each run, a new topology is generated and new source/sink nodes are randomly chosen. We calculate for each average node degree the average path stretch over the first 1000 messages. In all cases, the path stretch converges to one. The average path stretch is greater than one as the first messages follow long paths. The greater the average path stretch, the slower the convergence.

Table 1 suggests that, although higher density networks tend to converge faster, the impact is negligible.

Impact of network size. Nodes acquire their height from the sink node outward; the size of the network in number of hops hence influences convergence speed: the more hops the furthest node is from the sink, the longer the network takes to converge. We simulate topologies with different radii (i.e. the maximum number of hops between a node and the sink), and measure the average path stretch of the first 1000 messages. Simulation results are

average degree	average stretch
7	1.13 (+2.72%)
10	1.11 (+0.90%)
15	1.10 (baseline)

Table 1. Average node degree has a very limited impact on speed of convergence.

# of nodes	average stretch
100	1.05 (baseline)
200	1.11 (+5.71%)
300	1.18 (+12.4%)

Table 2. Network size has a limited impact on convergence speed.

Description	value
Number of nodes	86 nodes (68 indoors, 18 outdoors)
Deployment area	140 by 250 meters
Transmission power	-5 dBm
Battery capacity	11.6 kJ
Data generation rate	one packet every 7.5 min per node

Table 3. Parameters used in the Sense&Sensitivity experiment.

averaged over 1000 independent runs.

Table 2 shows that, although smaller networks are quicker to converge, the impact of network size on converge speed is limited.

3.4. Traditional vs. Reactive Gradient Routing

Our Reactive Gradient Routing scheme differs from traditional Gradient Routing:

- Traditional gradient routing uses an initialization phase, in which the sink node broadcasts a gradient setup message propagating throughout the network. This is often impractical. As links constantly evolve, the gradient is rapidly outdated at the initialization phase needs to be rerun. Periodically reconstructing the gradient is time- and energy-consuming. Reactive Gradient Routing does not use an initialization phase and is **entirely self-healing**, constantly reflecting topological changes by adjusting the node’s virtual heights.
- Reactive Gradient Routing requires **no signaling messages**. Moreover, MAC and routing layers interact closely, forming an **energy-efficient communication architecture**.

The application of the proposed solution to real-world testbed rollouts is described in the subsequent section.

4. The Sense&Sensitivity Experiment

The Sense&Sensitivity experiment mimics an Urban WSN [16]. 86 static WSN430 nodes are deployed in a large space, each periodically reporting light and temperature readings to one of the sink nodes. Measurements and network status information sent along with those measurements are collected from the different sink nodes into a central server[†]. Each node is driven by an MSP430 microcontroller and a CC1100 radio chip communicating on the 868 MHz European ISM frequency band. Table 3 details the parameters used in the experiment.

4.1. Experimental Results

Each node regularly generates a message. Inter-message duration is uniformly distributed in a 5-10min window; on average, a node generates a message every 7.5min. With 86 nodes deployed, the sink node receives a message on average every 5 s. 720 messages are thus collected per hour, or 17200 per day. The data contained in the messages is stored in a database for

[†] Data available at <http://senseandsensitivity.rd.francetelecom.com/>.

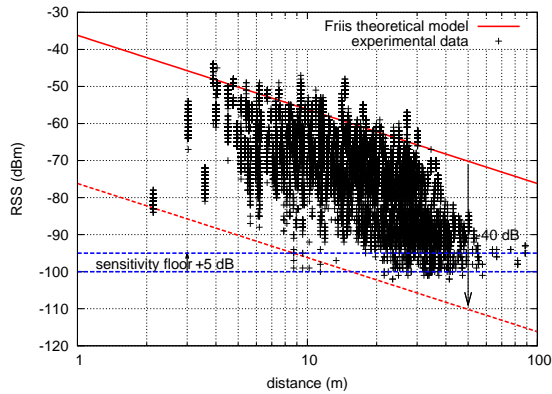


Figure 4. Relationship between RSS and distance obtained by experiment indoors.

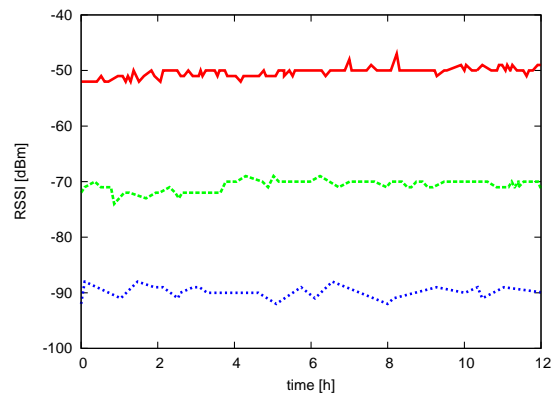


Figure 5. RSS variation over time for three representative links.

later inspection. We present results in a bottom-up approach, starting with deployment and energy consumption, before going up the communication layers: propagation, MAC, routing and application.

Deployment. The simple reactive gradient routing scheme enables a true deploy-and-forget experience. Once programmed and switched on, the 86 nodes are put in a box and scattered on the premises. Out of the 86, 68 are deployed indoors and 18 outdoors, over an area of roughly $140\text{ m} \times 250\text{ m}$ (Fig. 7). Adding or removing sensor and sink nodes does not require any (re)configuration.

Energy consumption. We mount a 1Ω resistor in series with the battery and visualize on an oscilloscope the voltage across that resistor. This is an image of the current flowing to the complete board. With its radio off, the node consumes $150\mu\text{A}$. Before sampling the channel for 2ms , the node needs 1.2ms to start and calibrate the radio. One such samples costs about 0.15mJ . As this happens every 128ms , as long as the network sits idle, the node consumes about 1mJ/s , i.e. 1mW . For receiving a message, a node consumes about 1mJ . When transmitting a packet, a node sends a series of microframes. Because the transmitting node's radio is always active between the first micro-frame and the end of the data transmission, it consumes about 9.5mJ to send a packet. Under the load above, these measurements indicate that the network has an expected lifetime of about 4 months.

RSS vs. distance. For each packet exchanged on the network, we measure and store its Received Signal Strength (RSS) value. We plot the RSS as a function of distance between nodes in Fig. 4. The external interference and multi-path fading effects of the mostly indoor deployment cause the RSS values to follow a Gaussian distribution when RSS is in dB (or a lognormal distribution when RSS is in mW).

Link stability. In Fig.5 we show the variation of the RSS over time. The standard deviation of the RSS readings for each of the 120 bidirectional links is 1.04 dB, indicating that link stability does not change much over time. That is, a weak link remains weak over time, as does a strong link.

Neighborhood evolution. Because some links are weak, a node does not always discover the same number of neighbors, causing the degree of a node to vary significantly. Fig. 6 shows the evolution of the node degree for three representative node, over a time span of 18 hours.

Neighborhood evolution is a natural consequence of the propagation characteristics described earlier, and is often neglected when designing routing protocols. Although it is possible to filter out neighbors with bad links to increase graph stability, this may artificially remove links which ensure network connectivity.

Connectivity graph. A snapshot of the connectivity graph is presented in Fig. 7. Orange lines interconnect neighbor nodes, the stronger those lines, the stronger the link. Neighbor reports are non-reciprocal when the orange line stops halfway between two nodes. The green line (lower right) represents the path taken by the last packet. Each node has 4.8 neighbors on average.

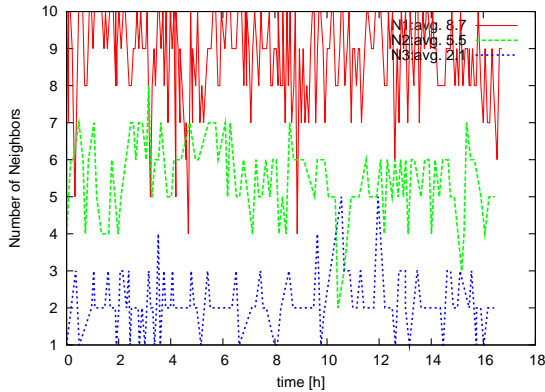


Figure 6. The node degree evolves significantly over time. These 3 nodes are representative of the whole network.

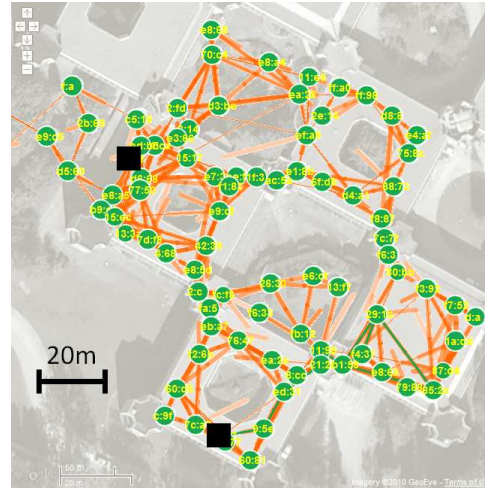
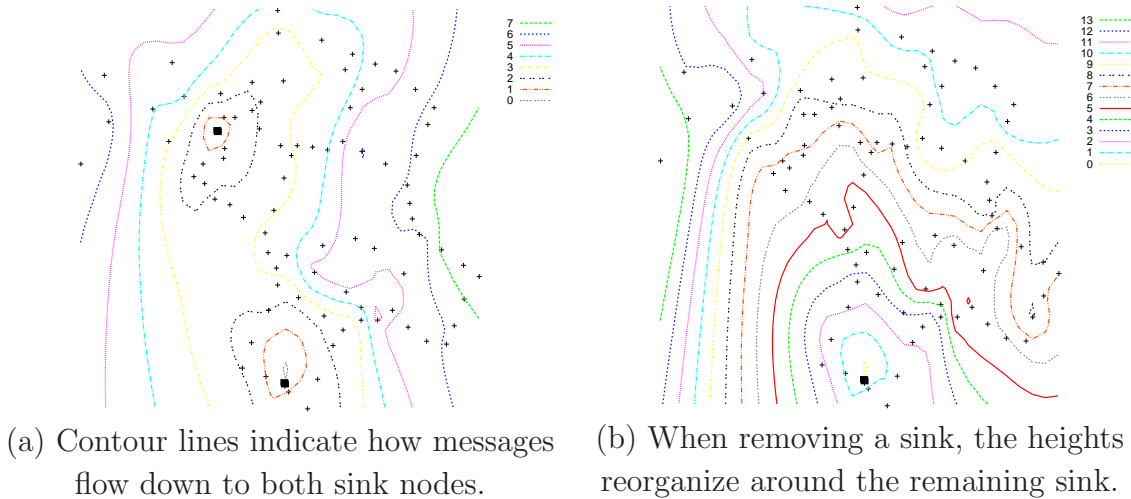


Figure 7. Bird's view of the connectivity graph. Circles represent nodes; squares represent sink nodes.



(a) Contour lines indicate how messages flow down to both sink nodes.

(b) When removing a sink, the heights reorganize around the remaining sink.

Figure 8. Contour lines show the virtual heights. '+' signs represent nodes; black squares represent sinks.

The premises used for the experiment are a number of interconnected square buildings. The outside walls of these buildings are entirely covered by windows which are coated with a thin layer of metal to reflect sunlight. Electromagnetic signals have a very hard time traversing this coating, which is the reason why the majority of the links leaving a building do so through the junctions.

The 7 nodes placed on the western-most part are connected to the network by a few very poor links. Although only a subset of the sent packets are eventually received by the sink nodes, if weak links were filtered out, these nodes would have been disconnected.

Average hop count and height. The routing protocol needs to find a path on the connectivity graph between each node and one of the sink nodes. By default, we deploy two sink nodes; a node implicitly sends data to the topologically closest sink. The contour lines overlaid on Fig. 8 (a) shows how the height values increase as we move away from either of the two sink nodes.

Data from nodes on the northern part flow to the northern sink; the rest flow to the southern sink. As there are more nodes in the northern part, the northern sink receives more

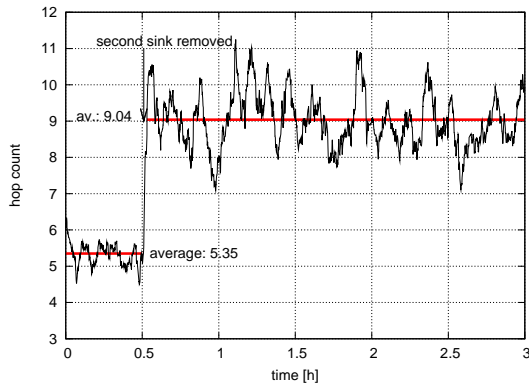


Figure 9. Hop count evolving after a sink node has been removed.

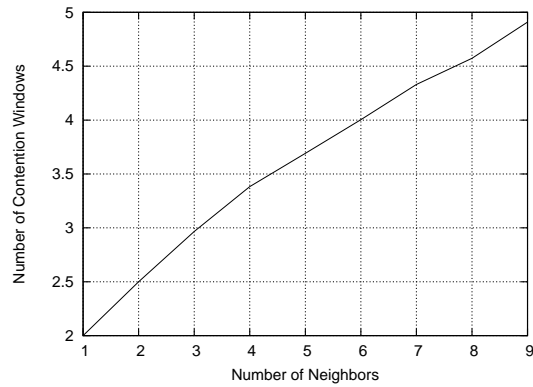


Figure 10. The more neighbors, the more contention windows are needed.

traffic. In the last 18 hours of the experiment, 18895 messages were received, 13823 (73.2%) by the northern sink and 5072 (26.8%) by the southern sink.

The sinks receive one message every 5s on average. As these messages come from different nodes, they travel different numbers of hops. The average hop count is 5.35 with two sink nodes.

End-to-end Latency. We call end-to-end latency the amount of time it takes for a packet to travel from its source to the sink, over multiple hops. Because nodes do not wait between the moment they receive a packet and the moment they relay it, latency is the average one-hop latency, multiplied by the average number of hops. The one-hop latency depends on the number of contention windows (CW) required at the MAC layer. Fig. 10 shows how that number changes with the number of neighbors; it presents experimental data averaged over all received packet.

A node has, on average, 4.43 neighbors. It will hence require 3.5 contention windows to send a packet. With a preamble duration of 129ms and a contention window duration of 12.8ms, this means that a single hop takes 173.8ms. In the topology with two sinks, depicted in Fig. 7, the end-to-end latency is 930ms.

Removing the second sink node. Reactive Gradient Routing copes with topological changes by constantly adjusting the heights of the nodes. One extreme case of topological change is when a sink node is removed; the heights of the node rearrange around the remaining sink(s). A secondary effect is that the average path length increases, as nodes which previously reported to the removed sink now need to send their messages to another sink farther away.

Fig. 9 shows the dynamics of the network when a sink is removed, by depicting the average hop count experienced by the packets received by the sink nodes. Hop count jumps from an average 5.35 to 9.04 because the sink is, on average, farther away. The figure also shows that the network self-organizes around the remaining sink node in a matter of minutes. Fig. 8 (b) shows how the virtual heights have reorganized around the remaining sink node. The maximum virtual height is now 13.

Adding a third sink node. Starting from the topology depicted in Fig. 7 (with two sink nodes), we add a third sink, again without any reconfiguration. The network adapts to this topological change, and the average hop count drops from 5.35 to 4.56.

5. Concluding Remarks

Analysis, simulation and experimental results have shown the functionality of Reactive Gradient Routing under real-world constraints.

Opportunities. We believe this solution to be suitable for convergecast WSNs, in which sinks are undifferentiated. As stressed by the experimental results, the solution is extremely robust against topological changes. Those changes encompass links or nodes (dis)appearing. This allows the network to run autonomously, a feature which gains importance as the number of nodes grows and node maintenance becomes highly impractical.

Reactive Gradient Routing passes the extreme test of adding and removing both nodes and sink nodes. We believe this approach to be attractive because it gives the user a true deploy-and-forget experience, while remaining energy-efficient.

Fig. 1 shows how multiple sinks can be used: a message flows to the topologically closest sink node. This behavior is very useful in applications where sink nodes collect undifferentiated data messages, as in the Sense&Sensitivity experiment. Additional sink nodes can be placed in the network in order to lower the average hop count and energy consumption without requiring extra configuration.

Limitations. From an application point of view, Reactive Gradient Routing has the following limitations. The solution assumes a fully convergecast traffic flow, where data messages reach any of the undifferentiated sink nodes. This has several implications. First, any-to-any communication is not possible, as such. It can be emulated by creating one set of heights per destination node, but this becomes inefficient as the number of destinations becomes closer to the number of nodes. Second, as such, a sink node is not capable of initiating messages towards a node in the network. Mechanisms such as source routing can be added to overcome this problem [10].

The MAC protocol we proposed is based on preamble sampling, which is best applicable to networks with low loads. As a consequence, the solution we propose will be sub-optimal at high loads, or in the presence of bursty traffic. Nevertheless, Gradient Routing does not preclude the use of different MAC protocols, such as synchronized solutions.

Future Work. Since nodes may have weak links to potential neighbor nodes, the discovered neighborhood can significantly vary over time. Although the routing is robust enough to adapt to these continuous topological changes, improvements can be made. Methods such as link hysteresis could be used to stabilize neighborhood. This should be done intelligently to avoid removing weak links which are needed to keep the network connected. This is an important issue which is worth investigating in future work.

References

- [1] C. Perkins, E. Belding-Royer, and S. Das. Ad hoc On-Demand Distance Vector (AODV) Routing, RFC3561. Technical report, IETF, July 2003.
- [2] D. Johnson, Y. Hu, and D. Maltz. The Dynamic Source Routing Protocol (DSR) for Mobile Ad Hoc Networks for IPv4, February 2007.
- [3] Javed Faruque, Konstantinos Psounis, and Ahmed Helmy. Analysis of Gradient-based Routing Protocols in Sensor Networks. In *International Conference on Distributed Computing in Sensor Systems (DCOSS)*, pages 258–275, Marina del Rey, CA, USA, June 2005. IEEE/ACM.
- [4] Olivier Powell, Aubin Jarry, Pierre Leone, and Jos Rolim. Gradient Based Routing in Wireless Sensor Networks: a Mixed Strategy. Technical report, arXiv.org automated e-print archives, Report CS-0511083, 2005.
- [5] Fan Ye, Gary Zhong, Songwu Lu, and Lixia Zhang. GRAdient Broadcast: A Robust Data Delivery Protocol for Large Scale Sensor Networks. *ACM Wireless Networks*, 11:285–298, May 2005.
- [6] Ke Liu and Nael Abu-Ghazaleh. Aligned Virtual Coordinates for Greedy Routing in WSNs. In *International Conference on Mobile Adhoc and Sensor Systems (MASS)*, pages 377–386, Vancouver, Canada, October 9-12 2006. IEEE.
- [7] Christos H. Papadimitriou and David Ratajczak. *On a Conjecture Related to Geometric Routing (Algorithmic Aspects of Wireless Sensor Networks)*, volume 3121/2004, chapter On a Conjecture Related to Geometric Routing, pages 9–17. Springer Berlin / Heidelberg, June 2004.
- [8] Rodrigo Fonseca, Sylvia Ratnasamy, Jerry Zhao, Cheng Tien Ee, David Culler, Scott Shenker, and Ion Stoica. Beacon Vector Routing: Scalable Point-to-Point Routing in Wireless Sensor networks. In *2nd Symposium on Networked Systems Design & Implementation (NSDI)*, pages 329–342, Boston, MA, USA, 2-5 May 2005. ACM.
- [9] Yao Zhao, Yan Chen, Bo Li, and Qian Zhang. Hop ID: A Virtual Coordinate based Routing for Sparse Mobile Ad Hoc Networks. *IEEE Transaction on Mobile Computing*, 6(9):1075–1089, September 2007.

- [10] RPL: IPv6 Routing Protocol for Low power and Lossy Networks, 8 March 2010. draft-ietf-roll-rpl-07 [work in progress].
- [11] Omprakash Gnawali, Rodrigo Fonseca, Kyle Jamieson, David Moss, and Philip Levis. Collection Tree Protocol. In *Conference on Embedded Networked Sensor Systems (SenSys)*, Berkeley, CA, USA, 4-6 November 2009. ACM.
- [12] Philip Levis, Neil Patel, Culler; David, and Scott Shenker. Trickle: A Self-Regulating Algorithm for Code Propagation and Maintenance in Wireless Sensor Networks. In *Symposium on Networked Systems Design and Implementation (NSDI)*, San Francisco, CA, USA, 29-31 March 2004.
- [13] T. Watteyne, A. Bachir, M. Dohler, Dominique Barthel, and Isabelle 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 (IWVAN)*, pages 639–644, New York, NY, USA, June 2006. IEEE.
- [14] Thomas Watteyne. *Energy-Efficient Self-Organization for Wireless Sensor Networks*. PhD thesis, INSA Lyon, November 2008. number 2008-ISAL-0082.
- [15] Hannes Frey and Ivan Stojmenovic. On Delivery Guarantees of Face and Combined Greedy-Face Routing Algorithms in Ad Hoc and Sensor Networks. In *Twelfth ACM Annual International Conference on Mobile Computing and Networking (MobiCom)*, pages 390–401, Los Angeles, CA, USA, September 23-29 2006. ACM.
- [16] Mischa Dohler, Thomas Watteyne, Tim Winter, and Dominique Barthel. Routing Requirements for Urban Low-power and Lossy Networks, RFC5548, May 2009.