Performance of Geographic Multicast Approach on a Real-life Platform

Lucas Augusto de Araujo Marques Leao, Violeta Felea, Hervé Guyennet

Performance of Geographic Multicast Approach on a Real-life Platform

Lucas Leão, Violeta Felea, Hervé Guyennet
Univ. Bourgogne Franche-Comté, FEMTO-ST institute / CNRS
DISC, 16 route de Gray, 25030 Besançon, France
Email: {firstname.lastname}@femto-st.fr

***Abstract*—The execution of real-life experiments provides meaningful insights on the performance of wireless sensor network solutions. Along with simulations, it is an important tool in the process of validating a new protocol. Our study case is GeoM, a geographic multicast routing protocol. We make use of the Future Internet of Thing IoT-LAB infrastructure in Strasbourg as the platform for the real-life experiments conducted for our solution. Our real-life experiments show that GeoM improves the network performance, when several metrics - packet loss, maximum energy consumption and latency - are simultaneously considered, compared to similar scheme routing. For scalability purpose we study correlation between the real-life experiment results and the values obtained in simulations, which proves to be consistent.**

I. Introduction

A Wireless Sensor Network (WSN) can be described as a combination of hardware and software features, responsible for monitoring and collecting environmental data in order to serve a final application. It is composed of electronic devices with limited capacity in terms of processing, energy and radio range, organized in an ad-hoc manner. Deploying several sinks (in Multi-Sink WSN/MS-WSN) appeared as a solution for several challenges: scalability, reliability, latency and network lifetime optimizations.

For smart city applications, the information availability is an important requirement. The sensed data must be delivered to a number of specific destinations. We can cite a city traffic control system as an application where information availability is fundamental. In city traffic control, the current state of the roads may be distributed over a number of sinks in order to control the timings in traffic lights based on the traffic flows. Another example of application is related to the tracking of buses in a public transportation system. The information of the position of a bus is relevant to a number of predefined bus stations. Sensor nodes are spread in the city tracking the passing buses. The data from each bus is forwarded to a number of predefined sinks, representing each bus station served by the concerned bus. In that sense, it is important to find mechanisms allowing the forwarding of the data to multiple destinations, such as k-anycast and multicast communication schemes. We can translate the application requirements into network capabilities. A MS-WSN solution must be capable of assuring network reliability, timeliness communication and optimized energy consumption.

Only few works in MS-WSNs focused on communication optimizations provide real-life results and none of them is concerned with k-anycast or multicast schemes. Moreover, most of the existing works in MS-WSN are oriented to simulation as validation method. In this work we analyze the behavior of an existing multicast communication scheme, the routing protocol GeoM [1], on top of the FIT IoT-LAB testbed [2]. We proposed GeoM as a first multicast scheme designed to address all the objectives: reliability, timeliness and network lifetime optimizations.

The remainder of this paper is organized as follows. Section II presents a brief discussion on MS-WSN works evaluated through real-life experiments. GeoM is presented in section III. The experimental scenarios and the discussion of the performance evaluation are presented in section IV. We conclude the paper in section V with the future perspectives.

II. Related Work

As presented in [1], the number of works capable of forwarding packets to all sinks using either multicast or k-anycast communication schemes is limited. Most of the solutions in MS-WSN deal with the forwarding of data to one sink, using unicast or 1-anycast communication schemes. When we narrow the filter to identify the works with real-life results, this number is even smaller. Only five solutions dealing with MS-WSN present real-life experiments, and none of them considers multicast or k-anycast communication schemes, as shown in table I.

TABLE I: MS-WSN solutions with real-life experiments

Comm Scheme	Hardware	OS	Sinks/Nodes	Ref.
Unicast	MICA2	TinyOS	3/11	[3]
Unicast	TelosB TmoteSky MicaZ	TinyOS	1/14	[4]
Unicast	SkyMote	Contiki	2/56	[5]
1-anycast	eZ430-RF2500T	–	1/16	[6]
1-anycast	VNODE	VROS	6/30	[7]

Solutions are either based on MAC scheduling [3], [4] or on routing [5]–[7] and are briefly described below.

The work in [3] is a MAC protocol. The objective is to define a duty cycled medium access strategy based on the routing tree and able to provide real-time communication in a MS-WSN. The information is sent directly to the sink in a staggered communication process. The nodes are synchronized and the packet is raised to upper levels until the sink is reached (synchronous skewed wakeup). The solution was fully validated with real-life experiments, and it considered scenarios with three sinks.

In [4] the authors define a MAC protocol that creates a very low duty cycle schedule based on the routing tree able to extend the network lifetime and assure low latency in comparison to other LPL (Low Power Listening) [8] approaches. The solution takes a sampling period as input and defines an adapted time schedule for the nodes to cooperate in the collection task. The solution was validated with real-life experiments, but it considered the existence of only one sink.

In [5] the objective is to adapt the existing RPL (Routing Protocol for Low-power and lossy networks) to a Multi-Sink WSN in order to reduce the average hop count, and consequently the energy consumption and packet loss. It defines a virtual root, which acts as a ”master sink” unifying all sinks into a single DODAG (Destination Oriented Directed Acyclic Graph). The solution was validated considering real-life experiments with two sinks.

The objective in [6] is to define a routing protocol for rechargeable MS-WSN in order to reduce network latency and optimize the energy consumption. The strategy defines a duty cycle-aware routing protocol, that opportunistically choses the next hop based on the latency to the best sink and the duty cycle. The authors describe configurations for real-life experiments considering only one sink. The presented results account only for the remaining energy.

In [7] the objective is to balance the network load by distributing the loads among the sink neighbors and thus prolonging the network lifetime. The algorithm makes use of the round-robin scheduling process to distribute packets among all sinks neighbors, in order to avoid the early depletion of energy and collisions in the sink vicinity. The solution was fully validated with real-life experiments, considering a network with six sinks.

For all cited solutions, the real-life deployments were fixed, with no variation of the nodes' position. The size of the networks was also reduced, with a maximum of 56 nodes. For the operating systems, the studied papers proposed experiments using TinyOS [9], Contiki [10] and VROS [11]. All OS are light-weighted operating systems for wireless sensor nodes. In terms of hardware, both MICA2 (MPR400CB) [12] and MICAz (MPR2400) [13] belong to the same family of motes, based on the Atmel ATmega128L and compatible with TinyOS. The motes TelosB [14], TmoteSky [15] and SkyMote [10] are based on the Texas Instrument MSP430, compliant with IEEE 802.15.4, also compatible with TinyOS and Contiki [10]. VNODE [11] is based on the Atmel ATmega128L, designed by VRLab, compatible with VROS and TinyOS.

The main objective of this work is to investigate the feasibility of GeoM routing protocol in real-life environments. We intend to analyze its performance and confront the real-life results against simulations. This comparison is on one hand part of the feasibility study and on the other, a validation of the previous simulation results related to the scalability problem.

As consequence of this objective, we contribute to existing implementations of real-life experiments for the multicast scheme. GeoM is the first multicast protocol to be implemented and tested on a real testbed sensor network environment. We chose an existing MS-WSN routing protocol, KanGuRou [16] to be part of the comparison, as no other multicast scheme with the same objectives was available. Being a k-anycast scheme (but responding to multicast needs when k equals the number of predefined addressed sinks), existing works in the field of experiments of sensor protocols is enriched with a first k-anycast scheme routing solution.

III. Geographic Multicast Routing

GeoM is a geographic routing protocol for MS-WSN that is capable of forwarding a generated packet to all available sinks in the network, while reducing the latency and balancing the energy consumption. Three hypotheses are done in geographic-based protocols, which are built on the basis of cartesian coordinates of nodes. Every node knows its own position, those of sinks and every node may obtain its neighbors' positions. Geographic-based protocols are tributary of dead-ends because of void areas. A node is in a void area if each of its neighbors is farther to a given sink than itself.

GeoM is divided into three phases: filtering, selection and forwarding. The first phase is responsible for filtering the neighbor nodes in order to create a list of candidates. During this phase, the neighbor nodes with negative progress or in void areas are eliminated. If during the filtering phase no valid neighbor is detected (because of void areas), the algorithm triggers a recovery mode with a different routing method. The second phase is dedicated to effectively selecting the forwarding candidates based on the calculated weighted metrics. The final phase is the actual routing, responsible for choosing the forwarder among the final candidates and forwarding the packet. Our algorithm does not require much knowledge of the network topology (no more than traditional geographic routing approaches). However, it uses broadcast messages to periodically advertise the existence of neighbor nodes, their void area status and the amount of consumed energy.

A. Filtering Process

The filtering starts at the current node having the new packet, as seen in figure 1. The current node verifies if it has a positive progress in relation to the target sinks, compared to the progress of the previous hop. If the current node presents a negative progress, it means that the packet was in recovery mode and no positive progress was yet found. The recovery strategy is based on the face routing and follows the same principles of the work in [16]. In this case, only one node is selected and the packet is directly forwarded to it.

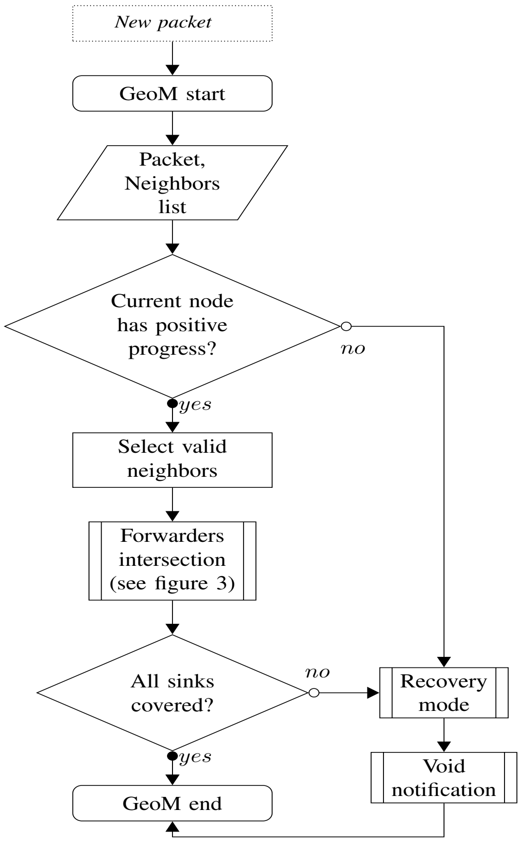


Fig. 1: GeoM flowchart

For the case a positive progress is detected, the current node goes through each neighbor searching for valid candidates. It eliminates the nodes with negative progress and neighbors in void areas, as exemplified in figure 2. The output of this phase is a structure containing the filtered neighbors per sink, which is passed to the forwarders intersection function, for the selection phase.

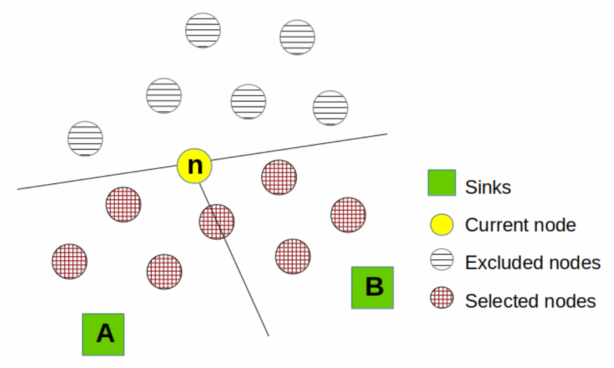


Fig. 2: Filtering process

If it was not possible to cover all sinks with at least one candidate, the recovery mode is triggered and a neighbor is selected using the void handling technique. At the same time, a broadcast message is sent to inform the neighbor nodes that the current node is in a void area.

B. Selection Process

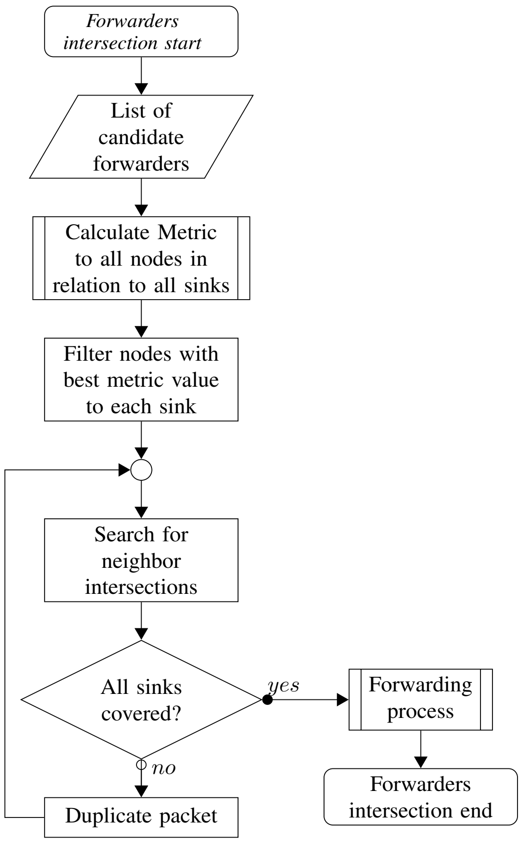
The selection phase is responsible for selecting the candidates with the best weighted metric towards the maximum number of sinks in order to avoid duplications. The process starts with the weighted metrics calculation to all candidates in relation to the sinks, as seen in figure 3. The weighted decision metric w represents the value of the calculated weighted metrics to all candidates in relation to all available sinks.

The weighted metric calculation considers the process in equation (1). It is based on the euclidean distance between the neighbor node v_i and the sink s_j denoted by $D(v_i, s_j)$, the total consumed energy of the neighbor node v_i denoted by $C(v_i)$ and the energy cost for the transmission from the current node n to the neighbor node v_i denoted by $E(n, v_i)$.

$$w_{v_i,s_j} = \alpha \times D(v_i, s_j) + \beta \times C(v_i) + \delta \times E(n, v_i) \qquad (1)$$

α, β, δ are the weights for each metric, $\alpha + \beta + \delta = 1$.

We note that this metric can be computed locally on a node based on direct neighbor knowledge.



Forwarders intersection start
List of candidate forwarders
Calculate Metric to all nodes in relation to all sinks
Filter nodes with best metric value to each sink
Search for neighbor intersections
All sinks covered?
yes
no
Forwarding process
Duplicate packet
Forwarders intersection end

Fig. 3: Forwarders intersection flowchart

The algorithm removes the neighbor candidates presenting a bigger weighted metric than the mean in relation to all neighbors and one specific sink ($\overline{w}_{s_j}$ in table II).

TABLE II: Calculated w for each pair $[s_j, v_i]$

	v_1	v_2	v_3	v_4	
s_1	w_{s_1,v_1}	$\cancel{w_{s_1,v_2}}$	w_{s_1,v_3}	w_{s_1,v_4}	$\overline{w}_{s_1}$
s_2	w_{s_2,v_1}	$\cancel{w_{s_2,v_2}}$	$\cancel{w_{s_2,v_3}}$	w_{s_2,v_4}	$\overline{w}_{s_2}$
s_3	$\cancel{w_{s_3,v_1}}$	$\cancel{w_{s_3,v_2}}$	w_{s_j,v_3}	$\cancel{w_{s_j,v_4}}$	$\overline{w}_{s_3}$

The objective is to apply a filter to select the candidates with the best values for the weighted metric. As a result of this process, the set of selected nodes is reduced, as shown in figure 4.

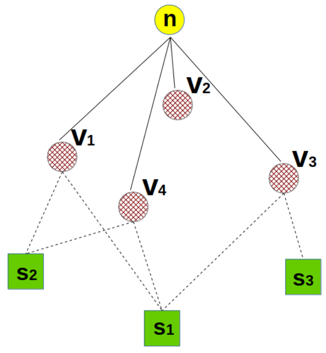


Fig. 4: Selection process

With the reduced list of candidates, the algorithm searches for the intersection of neighbors. The intersections are calculated based on the selection of sinks. The first selected sink is the closest to the current node, then the other sinks are selected based on the distance to the current node or to an already selected sink.

The objective of the intersection search is to avoid the packet duplication towards different paths. If a group of sinks shares a common forwarder candidate, the algorithm favors the selection of this neighbor node in order to reduce the number of packets circulating in the network.

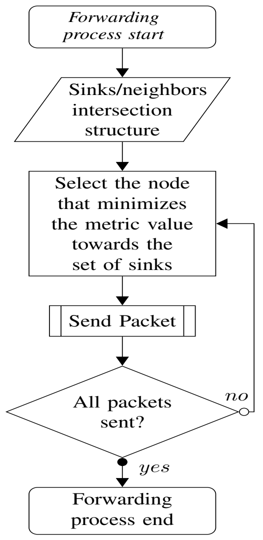
Considering the example, sinks $[s_1, s_2]$ share two common candidates $[v_1, v_4]$, sinks $[s_1, s_3]$ share one common candidate $[v_3]$, and sinks $[s_2, s_3]$ have no common candidates (see table III). In this example, the output is a structure containing the intersection of candidate forwarders and the list of sinks, such as $\{[s_1, s_2], [v_1, v_4]\}$ and $\{[s_3], [v_3]\}$.

TABLE III: Intersection search

Sinks	Candidate Forwarders			
s_1	v_1	-	v_3	v_4
s_2	v_1	-	-	v_4
s_3	-	-	v_3	-

C. Forwarding Process

The forwarding phase is the last process of the routing protocol and it is responsible for choosing the final forwarders, as displayed in figure 5. The selected forwarders are the neighbor nodes that minimize the mean of the weighted metric towards the maximum number of sinks. It takes as input the sinks/neighbors intersection structure and the table containing the calculated weighted metric. Then, it calculates the mean value of the calculated weighted metric in relation to a particular sink, as shown in table IV, where $\overline{w}_{v_1}$ is the mean of the weighted metric in relation to v_1.



Forwarding process start
Sinks/neighbors intersection structure
Select the node that minimizes the metric value towards the set of sinks
Send Packet
All packets sent?
no
yes
Forwarding process end

Fig. 5: Forwarding process flowchart

TABLE IV: Forwarder Selection

	v_1	v_2	v_3	v_4	
s_1	w_{s_1,v_1}	-	-	w_{s_1,v_4}	$\overline{w}_{s_1}$
s_2	w_{s_2,v_1}	-	-	w_{s_2,v_4}	$\overline{w}_{s_2}$
s_3	-	-	w_{s_3,v_3}	-	$\overline{w}_{s_3}$
	$\overline{w}_{v_1}$			$\overline{w}_{v_4}$	

Considering the sink/neighbors intersection structure as $\{[s_1, s_2], [v_1, v_4]\}$ and $\{[s_3], [v_3]\}$, for the entry $\{[s_1, s_2], [v_1, v_4]\}$, the forwarder neighbor is the node having $Min(\overline{w}_{v_1}, \overline{w}_{v_4})$, and for the entry $\{[s_3], [v_3]\}$, the forwarder neighbor is v_3.

Once the forwarders are selected, the actual routing takes place and the packet is sent to each forwarder with the corresponding list of sinks. It is at this moment that the packet is duplicated. Since the list of sinks is different for each forwarder, the packet must be split in order to be forwarded towards different directions.

IV. Evaluation

GeoM protocol was implemented and validated through simulations in [1]. A performance comparison was done with KanGuRou [16], another MS-WSN protocol capable of forwarding packets to all sinks. The simulation results demonstrated the viability of GeoM, with performance gains up to 54% in terms of maximum energy consumption. Now, our objective is to evaluate the performance of GeoM through real-life experiments, on top of FIT IoT-LAB testbed [2]. The FIT IoT-LAB is a large scale infrastructure (2071 wireless sensor nodes of different types) deployed over six locations in France. For the real-life experiments, we used the WSN430 Texas Instrument C1101 mote available in the FIT IoT-LAB testbed of Strasbourg, with a total deployment of 256 nodes, placed in a 3D-grid. The WSN430 has an MCU of 16-bit, with a ROM of 48KB and a RAM of 10KB [2].

We developed GeoM in C as an application module for Contiki OS [10], a lightweight open source operating system designed for limited resource devices. One of the advantages of the Contiki OS is its layered design, enabling the porting of a solution to different hardware devices. The original version of KanGuRou in [16] was implemented and tested with WSNet [17], an event-driven simulator, part of the WorldSens integrated environment for the design, development, prototyping and deployment of wireless sensor applications. We adapted and developed a version of KanGuRou, based on the original code, as an application module for Contiki OS.

The initial implementation of GeoM considered the COOJA mote, which is the native mote for the COOJA simulator. The COOJA mote simulates a sensor node, but it does not have the same restrictions of a real mote in terms of memory and processing capacity. Some adaptations were necessary in order to use GeoM and KanGuRou with the WSN430 mote.

TABLE V: Simulation and experimentation settings

	Cooja Simulator	FIT IoT-LAB Testbed
Network area	$270m \times 270m$	$10m \times 8m$
Deployment	Random grid	Grid
Executed networks	100	30/20
Mote type	Cooja Mote	WSN430 TI CC1101
Radio range/power	$50m$	$-10dBm$
Execution time	2 hours	30 minutes/6 hours
# of sensors	63 nodes	
# of sinks	5 nodes	
# of neighbors	8 nodes (maximum)	
Packet size	100 bytes	
Packet generation	20% chance at every minute for each sensor	
Radio type	802.15.4	
MAC Protocol	CX-MAC, modified version of [18]	

The performance evaluation comprises a comparison test with another routing protocol, KanGuRou [16], that was also implemented using Contiki OS, adapted to WSN430 and tested under the same conditions and configurations of GeoM. Table V summarizes the simulation and experimentation settings.

The evaluation considers the average of each metric for all executions, and the results are presented with a confidence interval of 95%. We analyze the latency, defined by the average time a packet takes to be routed from the source to sinks. Every sensor node is considered a source node and generates packets. The average latency is calculated for each network considering the sum of all latencies divided by the number of received packets. We also look at the hop count, which is the average number of hops from the source node to sinks. We analyze the maximum energy consumption, which concerns the node that consumed the largest amount of energy in the network at the end of the execution. It gives an indication of the network lifetime, since it shows how far the node is from the battery complete depletion. We consider a network to be alive as long as all nodes have some energy. Therefore, network lifetime is considered to be the earliest moment at which a node's battery is completely depleted.

A. *Real-life Results*

As presented in table V, we performed 30 executions of 30 minutes (short run) and 20 executions of 6 hours (long run) for each solution, which represents 270 hours of test execution in total. The nodes deployment in FIT IoT-Lab testbed form a grid based physical topology that produces a completely connected network, with a node being able to communicate with all the other nodes. However, since we wanted to validate the routing strategy, we needed to create longer paths. For that reason, we defined a different logical topology, by limiting the neighborhood to the nodes at the immediate physical proximity. The new logical topology created enough hop-distance among sensors and sinks to test the routing protocol.

Despite the fact that the nodes are fixed and the physical topology is the same, at runtime we may have different networks. The links among nodes vary from one execution to another due to interference and/or faulty nodes. This phenomenon results in slightly different networks, due to changes in the neighborhood. Figure 6 represents the nodes deployment with the sinks highlighted with circles, and the neighbor links for one particular execution.

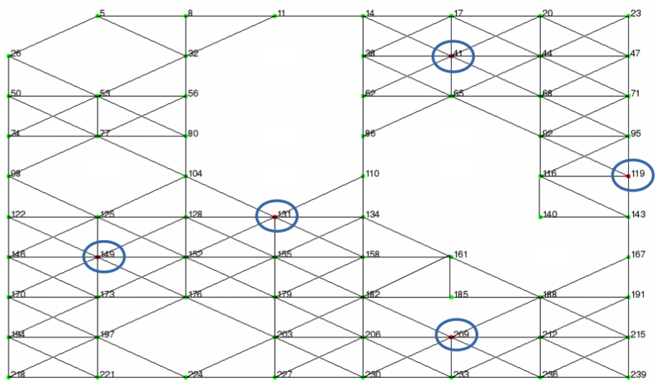


Fig. 6: Network topology (sinks are circled)

As we can see in figure 7, GeoM presents a higher maximum energy consumption compared to KanGuRou.

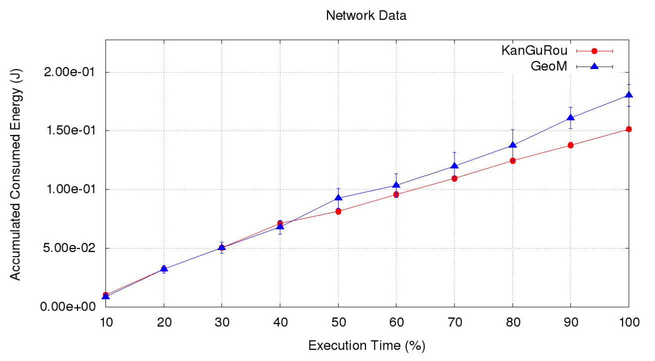
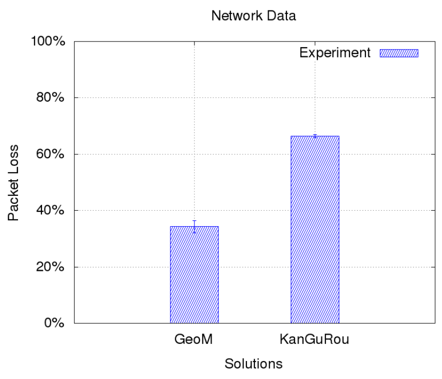
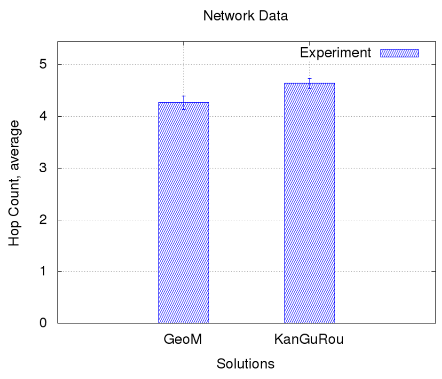


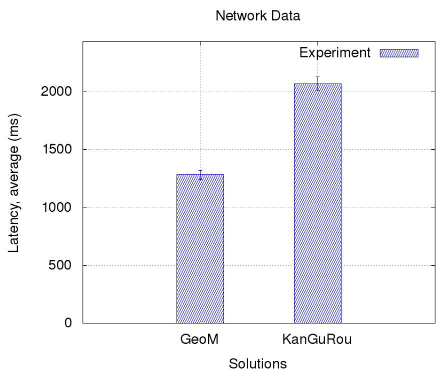
Fig. 7: Energy consumption over time for the node that consumed the maximum energy at the end of the execution (short run)



(a) Packet Loss



(b) Average Hop Count



(c) Average Latency

Fig. 8: Results of real-life experiments (short run)

This is the result of an elevated packet loss for KanGuRou. As displayed in figure 8(a), KanGuRou has a packet loss of roughly 70%, almost the double of the packet loss presented by GeoM. It means that for GeoM there are more packets being forwarded, and consequently a higher level of energy consumption. On the other hand, the average hop count for GeoM is slightly lower compared to KanGuRou, as seen in figure 8(b). It is explained by the fact that GeoM decides to duplicate the packets earlier in order to enable a fast progress towards the destination sinks, which also explains the latency results presented in figure 8(c). Due to important collision rate and presence of void areas, absolute latency is quite important. However, GeoM is able to reduce the average latency by approximately 38% compared to KanGuRou.

The huge difference in packet loss is also explained by the existence of faulty links, which heavily affects KanGuRou in comparison to GeoM. In this work we define a faulty link when communication between connected pairs cannot be accomplished due to channel interference, to physical incapacities of the sensors or to packet collisions. KanGuRou bases its routing decisions on the distance and the transmission energy cost. With this strategy, no path changes are performed. If faulty links are present in some routes, it is enough to drastically increase the packet loss. In GeoM, routes may change from one transmission to another, due to the metric related to the energy consumption. In the case of a faulty link, GeoM is less impacted. Moreover, since GeoM duplicates the packets slightly earlier, it means that each duplicated packet has a smaller group of destination sinks. As each generated packet must be delivered to all sinks, if the packet is lost before the duplication, the impact on the packet loss is much higher.

In order to have a better comparison point for the maximum energy consumption, we re-executed the experiments for GeoM and KanGuRou during an extended period of time (6 hours). The objective was to find a point of stability for the packet loss, so the maximum energy consumption could be analyzed under the same condition in terms of number of packets being forwarded. As we can see in figure 9(a), GeoM had a similar performance in terms of packet loss at the end of the 6-hour execution. The packet loss stabilized at approximately 60%. The maximum energy consumption results for GeoM and KanGuRou are almost equivalent, shown by Figure 9(b). We can see in Figure 10 that both solutions have a similar evolution of the energy consumption.

We can see in figure 9(c) that GeoM is able to reduce the average latency by approximately 67% (based on packets arrived to the sinks). As already mentioned, GeoM moves the packet towards the sinks in a faster way, since duplications take place slightly earlier, which translates to fewer hops, and consequently a reduction of the average latency.

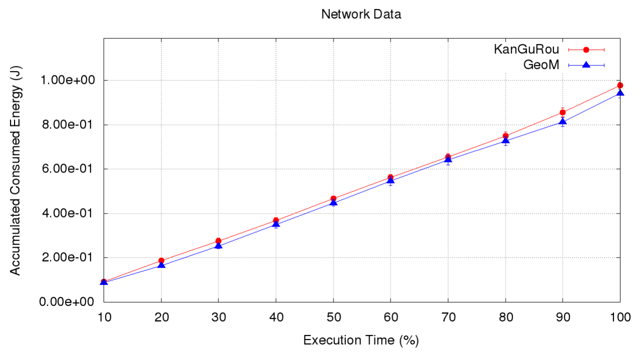
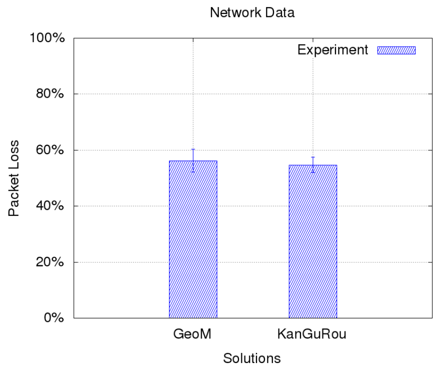


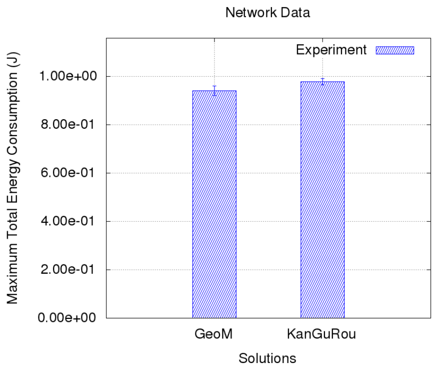
Fig. 10: Energy consumption over time for the node that consumed the maximum energy at the end of the execution (long run)

B. *Simulation Results*

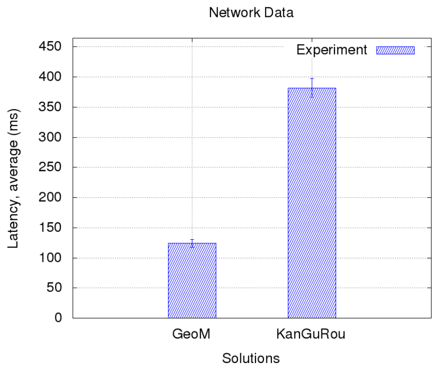
The real-life experiment is limited in terms of network size and physical deployment, even though our scenarios are larger than those in literature. Moreover, the real-life results show a marginal performance gain in terms of maximum energy consumption, which is mostly a consequence of the small size of the network and the regularity of the physical deployment. In that sense, it is important to identify a correlation between the real-life experiments and the simulations, in order to overcome the limitations and extrapolate the testing scenarios.



(a) Packet Loss



(b) Maximum energy consumption



(c) Average Latency

Fig. 9: Results of real-life experiments (long run)

To do so, we tried to create and execute simulation scenarios that would be close to the experiments.

For the simulations, 100 different networks were randomly generated. Figure 11 represents one randomly generated network. Similarly to the real-life experiment, each node has approximately 8 neighbors in its communication range. However, the link quality and interference level are not the same. The communication range is 50m with a bidirectional link and the interference range is of 100m. The simulation environment is much more stable compared to the real-life experiment. This stability can be evidenced by the packet loss results, as seen in figure 13(a). GeoM and KanGuRou share similar packet loss results for the simulation.

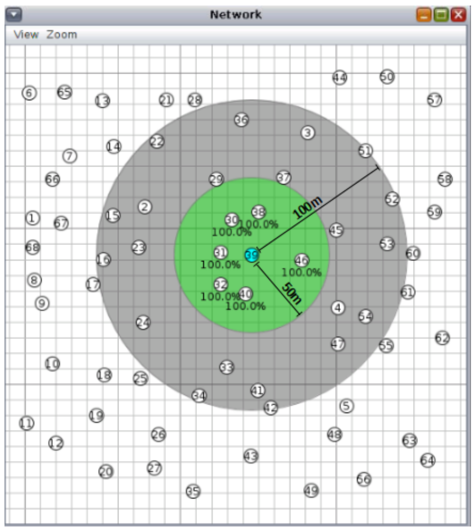


Fig. 11: Physical deployment for the simulation

Because of the stable environment, we are able to compare the maximum energy consumption. We can see in figure 12 that GeoM and KanGuRou have almost the same maximum energy consumption. These relative results are coherent with those from experiments (see figure 10). This is the expected behavior for a network with 63 nodes and 5 sinks. In [1] GeoM was validated with different network sizes, going from 50 up to 300 sensor nodes. It was noticed that GeoM presents much better performance in terms of maximum energy consumption for larger networks, with gains from 5% up to 26% for networks with void areas and 23% up to 54% for networks without void areas. However, the comparison with the real-life results allows us to conclude that the performance presented by GeoM is consistent and follows the same trend of the simulation.

The same comment applies for the average latency results, computed here on packets arrived to all sinks. We can see in figure 13(c) that GeoM and KanGuRou have similar results, with a difference of 3.3%. However, for the simulations presented in [1], with larger networks, we can observe better results for GeoM in terms of average latency, with performance gains of 10%. Nevertheless, we can see in figure 13(b) that the average hop count for GeoM is smaller compared to KanGuRou, which for the routing perspective results in lower latencies.

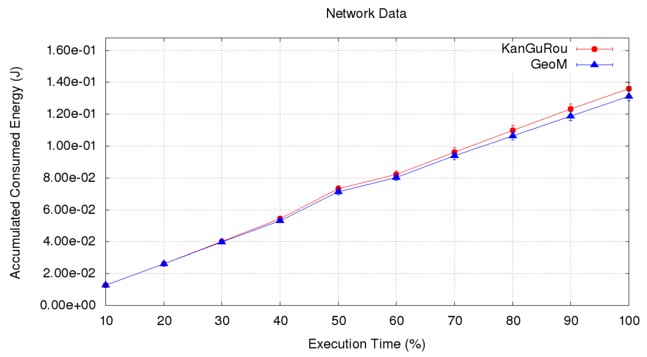
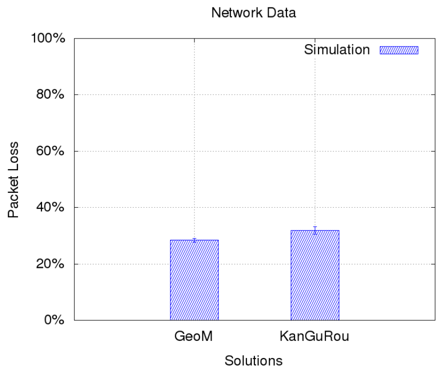


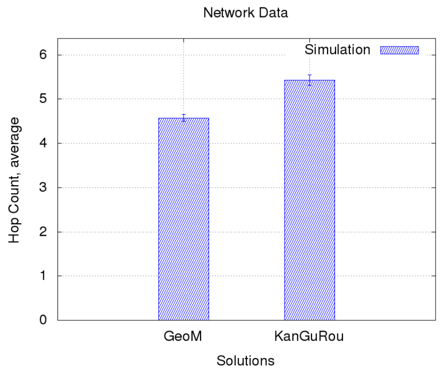
Fig. 12: Energy consumption over time for the node that consumed the maximum energy at the end of the simulation

V. Conclusion

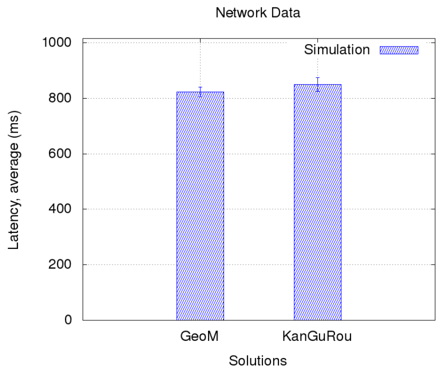
This paper presented the real-life results for the Geographic Multicast (GeoM) routing protocol as further analysis of its performance. The results were obtained with experiments



(a) Packet loss



(b) Average hop count



(c) Average latency

Fig. 13: Results of simulations

carried out within the FIT IoT-Lab platform and simulations using COOJA simulator.

The main conclusion of this work concerns the correlation between the real-life experiments and the simulations : the results are consistent and comparable, for both GeoM and KanGuRou solutions, analyzed under the same criteria and executed on the testbed in Strasbourg. A parallel can be made and the testing scenarios can be extrapolated as in the simulations performed in [1]. As for the comparative experiments, in short runs, GeoM offers better delivery rate and latency compared to KanGuRou. For long runs, the collision rate increases (more packets need to be delivered) and performance is similar in energy. Latency is packet loss dependant and experiments could not compute latency of packets towards all sinks. In simulations, smaller packet loss could be observed, so the latency is computed for packets delivered to all sinks.

As noticed with the experimental tests, the link quality, interference and collisions have a high effect on metrics such as latency and energy consumption. Because of that, we measure the importance of considering such aspects during the routing decision. The routing metrics must account for the link quality in order to avoid delays. As future work, we consider integrate the link quality aspect to the routing decision along with the other aggregated metrics.

Acknowledgment

This work has been partially supported by the Brazilian National Council for Scientific and Technological Development (CNPq). Computations for simulations have been performed on the supercomputer facilities of the Mésocentre de calcul de Franche-Comté.

References

[1] L. Leão and V. Felea, "Latency and network lifetime trade-off in geographic multicast routing for multi-sink wireless sensor networks," in *International Conference on Mobile, Secure and Programmable Networking*. Springer, 2018, pp. 1–12.

[2] F. I. T. Facility, "FIT IoT-LAB," https://www.iot-lab.info/, accessed: 2018-03-20.

[3] S. H. Lee and L. Choi, "SPEED-MAC: speedy and energy efficient data delivery MAC protocol for real-time sensor network applications," *Wireless Networks*, vol. 21, no. 3, pp. 883–898, 2015.

[4] U. M. Colesanti, S. Santini, and A. Vitaletti, "DISSense: An adaptive ultralow-power communication protocol for wireless sensor networks," in *International Conference on Distributed Computing in Sensor Systems and Workshops (DCOSS)*. IEEE, 2011, pp. 1–10.

[5] D. Carels, N. Derdaele, E. De Poorter, W. Vandenberghe, I. Moerman, and P. Demeester, "Support of multiple sinks via a virtual root for the RPL routing protocol," *EURASIP Journal on Wireless Communications and Networking*, pp. 1–23, 2014.

[6] D. Gao, Y. Liu, F. Zhang, and J. Song, "Anycast routing protocol for forest monitoring in rechargeable wireless sensor networks," *International Journal of Distributed Sensor Networks*, 2013.

[7] C. Wang and W. Wu, "A load-balance routing algorithm for multi-sink wireless sensor networks," in *International Conference on Communication Software and Networks (ICCSN)*. IEEE, 2009, pp. 380–384.

[8] J. Polastre, J. Hill, and D. Culler, "Versatile low power media access for wireless sensor networks," in *2nd International Conference on Embedded Networked Sensor Systems*. ACM, 2004, pp. 95–107.

[9] TinyOS Alliance, "TOSSIM," http://docs.tinyos.net/index.php/TOSSIM, accessed: 2017-05-10.

[10] Contiki OS, "The open source os for the internet of things," http://www.contiki-os.org/, accessed: 2018-03-20.

[11] C. Wang and W. Wu, "A light-weighted operating system with deadlock prevention strategy for wireless sensor nodes," in *WRI International Conference on Communications and Mobile Computing (CMC)*, vol. 1. IEEE, 2009, pp. 578–583.

[12] CROSSBOW, "MICA2 datasheet," accessed: 2017-05-10. [Online]. Available: https://www.eol.ucar.edu/isf/facilities/isa/internal/CrossBow/DataSheets/mica2.pdf

[13] Memsic, "MICAz datasheet," accessed: 2017-05-10. [Online]. Available: http://www.memsic.com/userfiles/files/Datasheets/WSN/micaz_datasheet-t.pdf

[14] ——, "TelosB datasheet," accessed: 2017-05-10. [Online]. Available: http://www.memsic.com/userfiles/files/Datasheets/WSN/telosb_datasheet.pdf

[15] M. Corporation, "TmoteSky datasheet," accessed: 2017-05-10. [Online]. Available: http://www.eecs.harvard.edu/~konrad/projects/shimmer/references/tmote-sky-datasheet.pdf

[16] N. Mitton, D. Simplot-Ryl, M.-E. Voge, and L. Zhang, "Energy efficient k-anycast routing in multi-sink wireless networks with guaranteed delivery," in *International Conference on Ad-Hoc Networks and Wireless*. Springer, 2012, pp. 385–398.

[17] A. Fraboulet, G. Chelius, and E. Fleury, "Worldsens: development and prototyping tools for application specific wireless sensors networks," in *6th International Symposium on Information Processing in Sensor Networks (IPSN)*. IEEE, 2007, pp. 176–185.

[18] M. Buettner, G. V. Yee, E. Anderson, and R. Han, "X-MAC: a short preamble MAC protocol for duty-cycled wireless sensor networks," in *Proceedings of the 4th International Conference on Embedded Networked Sensor Systems*. ACM, 2006, pp. 307–320.