Node Ranking in Wireless Sensor Networks with Linear Topology

Rodrigue Domga Komguem, Razvan Stanica, Maurice Tchuente, Fabrice Valois

Node Ranking in Wireless Sensor Networks with Linear Topology

Rodrigue Domga Komguem*†‡, Razvan Stanica*, Maurice Tchuente†‡, Fabrice Valois*
*Université de Lyon, INSA Lyon, CITI-Inria, F-69621, Villeurbanne, France
†Université de Yaoundé I, CETIC, LIRIMA, Faculté des Sciences, BP 812, Yaoundé, Cameroun
‡Sorbonne Universités, UPMC Univ Paris 06, UMMISCO-IRD, F-93143, Bondy, France
rodrigue.domga-komguem@insa-lyon.fr

***Abstract*—In wireless sensor networks with linear topology, knowing the physical order in which nodes are deployed is useful not only for the target application, but also to some network services, like routing or data aggregation. Considering the limited resources of sensor nodes, the design of autonomous protocols to find this order is a challenging topic. In this paper, we propose a distributed and iterative centroid-based algorithm to address this problem. At each iteration, the algorithm selects two virtual anchors and finds the order of a subset of nodes, placed between these two anchors. The proposed algorithm requires local node connectivity knowledge and the identifier of the first sensor node of the network, which is the only manually configured parameter. This solution, scalable and lightweight from the deployment and maintenance point of view, is shown to be robust to connectivity degradation, correctly ordering more than 95% of the nodes, even under very low connectivity conditions**

Keywords*—*Linear Wireless Sensor Networks, Nodes ranking, Self-Configuration, Centroid-Based Localization.

I. Introduction

The topology and architecture of wireless sensor networks (WSN) generally depend on the target application and the geographical area in which nodes are deployed. A linear WSN (LWSN) is a special case, where the physical topology of the network is a line [1]. The applications of LWSN are diverse, e.g. large infrastructure monitoring, such as bridges [2] and dams [3], road traffic observation [4], or border control [5]. Node localization information is needed in LWSN not only for network operations (e.g. geographic routing and data collection mechanisms), but also at the application layer: if a problem is detected by one of the few hundred sensors monitoring a 10 km long bridge, node location information is required for an intervention. While precise coordinates (in the range of cm) might be required in some scenarios, a relative *ranking* information is often enough for most LWSN applications. Considering a deployed LWSN, a sensor's rank is defined as the number of nodes closer to the origin compared with that sensor. In this paper, we are interested in autonomous solutions for node ranking in LWSN; indeed, manually assigning coordinates, or ranks, during the node deployment phase is an error prone, time consuming operation, also requiring qualified workforce.

In general, the evaluation metric for localization algorithms is the localization error, i.e. the distance between the real and the estimated coordinates. However, our focus is on node ranking, a less precise, but usually sufficient localization information. Theoretically, the received signal strength indicator (RSSI) is a decreasing function of the distance, an assumption largely used for node ranking in a LWSN [10]–[13]. Nevertheless, multiple studies show that, because of the hardware quality or constraints related to the environment (e.g. obstacles and climate conditions), the RSSI by itself does not provide enough information for correct range estimation. Rather than using information provided by the hardware, Lotker *et al.* [14] use neighborhood information to rank nodes in a LWSN. The authors propose a centroid-based approach, in which they assume that the first and the last node of the network are anchors with known positions. In the proposed distributed algorithm, they also assume that there is a global clock for node synchronization. Inspired by [14], we prove in this paper that it is possible to rank nodes in a LWSN knowing only the first node of the network, relaxing the two-anchors constraint. Based on this result, we propose an iterative algorithm for node ranking in a LWSN where, in each iteration, a virtual anchor is selected, allowing to correctly rank more and more nodes in the network. Moreover, unlike the solution proposed in [14], our iterative algorithm does not require any global clock for node synchronization. To understand the impact of parameters on system performance when a LWSN is deployed and to find some guidelines for network deployment, we conduct a series of simulations to investigate the system behavior when using our centroid-based technique.

In the remainder of this paper, we start by formulating the node ranking problem in LWSN in Section II and continue with a discussion of related work in Section III. In Section IV we discuss the usage of a single anchor to rank nodes, an idea used in our iterative algorithm, presented in Section V. We evaluate the performance of the centroid-based algorithm in Section VI, before concluding in Section VII.

II. Problem formulation

We consider a network consisting of N sensors (also called "nodes" or "sensor nodes" in the following), $s_0, s_1, \ldots, s_{N-1}$, uniformly deployed along a line in this order, i.e s_{i-1} is the i$^{\text{th}}$ sensor of the network. We denote by $\mathcal{G} = (\mathcal{S}, \mathcal{L})$ the directed graph associated to the network. The vertices $\mathcal{S}$ of this graph correspond to the sensor nodes and there is an edge $(s_i, s_j) \in \mathcal{L}$ whenever s_j can receive messages from s_i. We denote by $\Gamma(s_j)$ the direct (or one-hop) neighbors of a sensor s_j, i.e all the nodes s_i such that $(s_i, s_j) \in \mathcal{L}$, while $h(s_i, s_j)$ is the number of hop between sensors s_i and s_j. We denote by $x_i \in \mathbb{R}$ the estimated one-dimensional coordinate of s_i. Sensor s_0, the first sensor of the line, is considered as the sink, with a well known coordinate $x_0 = 0$. In this paper, we denote by Δ

the degree of the sink s_0.

The inputs of our problem are the network connectivity graph $\mathcal{G}$ and the sink s_0. Given these inputs and a node s_i, we define the rank (or the position) of s_i as the number of sensors deployed between s_0 and s_i. The question is whether we can correctly rank each sensor in the network. Therefore, the output of the problem is a sequence of sensors, $s_{(0)}, s_{(1)}, \ldots, s_{(N-1)}$. An ideal solution for this problem would produce a node sequence which corresponds to the ground-truth, i.e. $s_{(1)} = s_1, \ldots, s_{(N-1)} = s_{N-1}$.

For performance evaluation, we use the ratio of correctly ranked pairs of nodes. For any pair of nodes (s_i, s_j), the sensors are deployed in a given order in the network: either s_i is located before s_j, in which case $i < j$, or the opposite. In the output sequence, the pair (s_i, s_j) is well ranked if the order of the two nodes is the same as in the real deployment. Therefore, the ratio of correctly ranked pairs of nodes, r, is defined as: $r = \frac{C_{ok}}{C}$, where C_{ok} is the number of well ranked pair of nodes and C is the total number of pairs of nodes in the network. A high value of r means an output sequence of good quality, with $r = 1$ for a perfect order. Because of the unpredictable and unstable nature of links in WSN, two problems can appear: the ranking of two nodes can be reversed, or a node might be entirely absent from the output sequence. The proposed metric is interesting since it accounts for both these problems.

III. Related work

Node localization is a related and widely studied subject in the field of WSN. Most localization solutions use ranging techniques based on measurements provided by the hardware: the angle of arrival (AoA), the time of flight (ToF), the RSSI [6], or the phase difference between transmitter and receiver [7]. A second class of localization approaches uses network topology [8] or neighborhood correlation [9] information to estimate the distance between nodes.

Regarding node ranking in LWSN, most of the existing solutions make the assumption that the strength of a radio signal decreases with the distance between the two communicating nodes, meaning that the closest neighbor always produces the highest RSSI value. In [10], the authors propose to randomly select a node and try, from it, to build a sequence containing all nodes. In [11], an ordered matrix is used instead. Each row of the matrix is for a node and contains the ordered list of neighbors of that node. The neighbors are ranked in the decreasing order of the measured RSSI. Such an ordered matrix avoids to test all the possible permutations, as in [10]. In [12], the first node is known and authors also propose mechanisms to handle exceptions, which occur when the output sequence does not contain all the nodes.

These simple RSSI-based approaches do not work very well in practice, where the radio signal between two nodes propagates over multiple paths, either constructive or destructive. Therefore, the idea of frequency diversity is exploited in [13], where nodes transmit and measure the RSSI on 16 different frequency channels. The authors propose to build a probability tree using the maximum RSSI measured on all these frequencies, the node ranking being given by the path with the maximum joint probability. However, the evaluation of this protocol on a testbed, during a working day, shows that surrounding WiFi networks have a significant negative impact on its performance.

Instead of using a metric provided by the hardware, a neighborhood-based solution is proposed in [14]. In this solution, the position of a node is estimated as the centroid relative to its neighbors, the idea being to bring physically connected neighbor nodes closer to one another in terms of virtual coordinates. The same idea, illustrated in Figure 1, is also exploited in [15], for the case of 2-dimensional networks. Assuming a simple 1-dimensional topology with only five nodes, as in Figure 1, the authors consider that the positions and the coordinates of nodes s_0 and s_4 are known: $x_0 = 0$ and $x_4 = d, d > 0$. The coordinates of nodes s_1, s_2 and s_3 are initially unknown, and set to 0. Nodes begin by discovering their neighbors (and the coordinates of these neighbors), building a connectivity graph, as in Figure 1a. Once the neighbors are known, each of the nodes s_1, s_2 and s_3 sets its coordinate to the average of its neighbors coordinates. We thus have $x_1 = 0, x_2 = d/4$ and $x_3 = d/3$, as shown in Figure 1b. These new coordinates are announced to the neighbors and, after a second computation, we have $x_0 < x_1 < x_2 < x_3 < x_4$, as shown in Figure 1c, which gives the correct rank of nodes in the network. In [15], nodes initialize their coordinates with random values instead of 0. The fact of using fixed or random initial coordinates does not affect the ranking performance of the centroid algorithm, but only its convergence time.

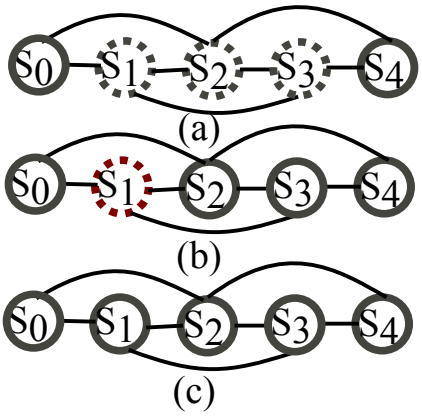

Fig. 1: The centroid-based algorithm proposed in [14]: (a) After neighbors discovery : the coordinates of nodes $s_1 - s_3$ are unknown (b) Nodes s_2 and s_3 calculate their coordinates (c) Node s_1 calculates its coordinate

As explained, in [14], sensors s_0 (the first node) and s_{N-1} (the last node) of the network have known coordinates and they are used as anchors. Moreover, all nodes are synchronized by a global clock, used to compute the beginning of each iteration. In each iteration, all the nodes broadcast their coordinates to their neighbors. The two anchors check whether all their neighbors have non-null coordinates and, when this is true, they broadcast a STOP message, which will be forwarded over the whole network in the following iterations. When a sensor s_i receives a STOP message from both anchors, it stops updating its coordinate. The coordinate of sensor s_i at iteration t, is given by:

$$x_i^t = \frac{\sum_{s_k \in \Gamma(s_i)} x_k^{t-1}}{|\Gamma(s_i)|} \quad (1)$$

Considering a unit disk graph (UDG) model for node connectivity, Lotker *et al.* [14] prove that, after $h(s_0, s_{N-1})+1$ iterations, their solution produces the correct order of nodes in the network. Two important observations can be made. First of all, this solution only produces the correct ranking; converging to stable values for the virtual coordinates might require extra iterations. Second, this result is only true for LWSN having a number of nodes larger than 2Δ. Indeed, if there are less nodes in the network, some of them will have the same neighborhood under the UDG assumption, and the algorithm will produce the same coordinate for these nodes, as we show in the next section.

IV. From two to one anchor solution

We are interested in energy-efficient, lightweight and self-configured protocols for node ranking in a LWSN, where the number of parameters manually configured during network deployment is minimized. Therefore, we begin by investigating the consequences of an incorrectly configured second anchor in the solution proposed by Lotker *et al.* [14]. However, we slightly modify the stopping condition, as described by Algorithm 1. In this modified algorithm, an anchor sends a STOP message when the coordinates of its neighbors converge to a stable value. As the coordinates of all non-anchor nodes are initialized to 0, it is easy to prove, by induction on t, that $\forall j, 0 < j < k, x_j^{t+1} \geq x_j^t$ and $x_j^t < x_k$, where s_k is the second anchor. These two conditions are sufficient to guarantee that the coordinates will converge to a value and the sinks will send the STOP messages.

Algorithm 1 Modified ranking algorithm for anchor s_a

1: Transmit the s_a coordinate (x_a)
2: **if** For All $s_i \in \Gamma(s_a), x_i^t - x_i^{t-1} < \epsilon$ **then**
3: Transmit $STOP_a$
4: **end if**

Using a dedicated linear network simulator, we evaluate the performance of the centroid-based approach when the second anchor is incorrectly defined and under varying communication range conditions. In a first step, we consider an UDG model, i.e two nodes are able to communicate if the distance between them is less than the communication range R. With this model, links between nodes are then symmetric. We consider a network of 21 nodes linearly and uniformly deployed with a fixed distance of 5 m between two consecutive sensors. Thus, sensors are deployed in the order $s_0, s_1, \ldots, s_{20}$. To take into account node degree, we vary R from 10 m to 100 m. This allows to have a network diameter range from 1 to 10 hops. For all simulations, s_0 is considered as the first anchor. The results are shown in Figure 2.

Figure 2a corresponds to a communication range of 25 m (i.e. $\Delta = 4$). In this configuration, sensors from s_1 to s_{20}, one after another, are setup as the second anchor. On this figure, the x-axis corresponds to the position of the second anchor, e.g. position 3 means that sensor s_3 is configured as the second anchor. The main message highlighted by this figure is that, when the sensor at the position k (sensor s_k) is configured as the second anchor, the estimated ranks of sensors $s_1, s_2, \ldots, s_{k-1}$ are always correct.

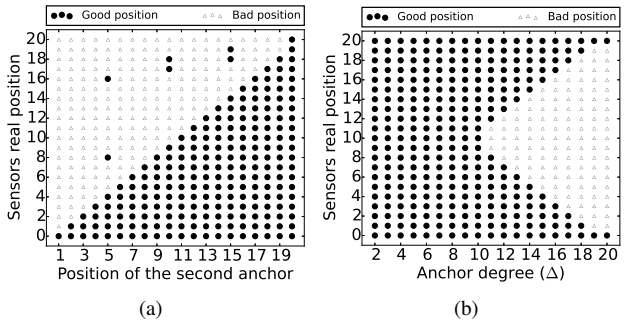

Fig. 2: Node ordering in a LWSN. The y-axis corresponds to the real sensor positions and a black point means that the estimated rank of the sensor is correct, while an unfilled triangle point means a wrong estimated rank. (a) Impact of the position of the second anchor ($\Delta = 4$) (b) Impact of node degree (node s_{20} is configured as the second anchor)

In Figure 2b, to investigate the impact of node degree, we vary Δ from 2 ($R = 10$ m) to 20 ($R = 100$ m) nodes, as indicated by the x-axis. In this configuration, s_0 and s_{20} are the two anchors. When Δ is less than 10 nodes, any two sensors have a different one-hop neighborhood, and the centroid-based approach gives perfect ranking results. By increasing Δ (i.e the communication range), sensors at the network center start having all the other nodes in their neighborhood. When $\Delta = 20$, the network becomes a clique and all the nodes have the same neighborhood. In such a situation, these nodes will have the same coordinates, and then the same rank. These nodes are the ones with a wrong estimated rank in Figure 2b.

The results presented in this section show that, considering an UDG connectivity model and a network of size $N \geq 2\Delta+1$, regardless the node s_k designated as the second anchor, the estimated ranks of sensors $s_1, s_2, \ldots, s_{k-1}$ are correct. In the next section, we describe a solution in which the nodes in the entire network can be iteratively ranked, without the precise knowledge of a second anchor. However, the results also indicate that the proposed solution only functions in networks with a diameter of at least 3 hops. Nevertheless, we argue that LWSN generally respect this constraint in practice, and we evaluate the performance of the proposed algorithm in realistic scenarios in Section VI.

V. Iterative Node Ranking Algorithm

In our quest of reducing the number of manually configured parameters in the network, we notice that, even when the second anchor is not perfectly chosen, a part of the nodes are still correctly ranked. Based on this observation, we propose, in this section, an iterative algorithm for node ranking in a LWSN. This algorithm runs in two steps: in the first step, each node discovers its neighborhood, while the second step, executed only by some well-selected nodes, finds the ranks of all the nodes in the network.

A. Neighborhood discovery

In this first step of our solution, nodes discover their one and two-hop neighbors. Simple *hello* messages can be broadcast for one-hop neighbor discovery. For two-hop neighbors discovery, nodes exchange their list of one-hop neighbors.

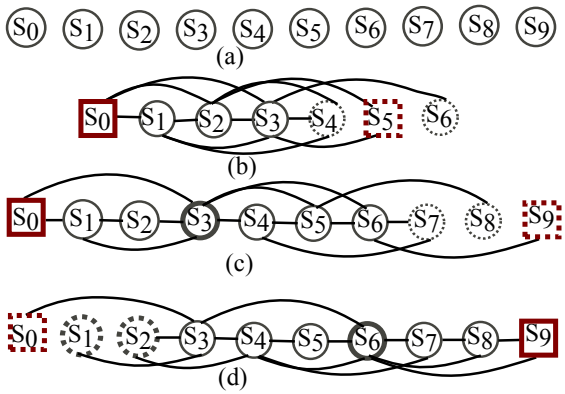

Fig. 3: An example of a 10-nodes LWSN, $\Delta = 3$. (a) Node deployment; (b) Local topology information at s_0; (c) Local topology information at s_3; (d) Local topology information at s_6. Dotted lines are for two-hop neighbors and red rectangles are for anchor nodes.

Considering a network of 10 nodes, uniformly deployed under an UDG connectivity model with $\Delta = 3$, as shown in Figure 3a, each node in the network builds a local view of the network. This local view is shown for s_0, s_3 and s_6 in Figures 3b, 3c and 3d, respectively.

Using this approach, each node in the network has a complete view of the connectivity between its one-hop neighbors. Concerning the two-hop neighbors, connectivity information is only partially available: the identity of the two-hop neighbors is known, as well as their relations with one-hop neighbors; however, connectivity information between two-hop neighbors is not known.

B. Node ranking

These local topologies, available at each node, are used for node ranking. As shown in Section IV, the nodes between two selected anchors are well ranked, as long as the distance between the two anchors is sufficient. Therefore, we propose to start with the first node, s_0, and rank the nodes in its one-hop neighborhood. The two anchors are then iteratively modified, selecting at each step nodes placed farther away, until coordinates are computed for the entire network. Practically, s_0 chooses one of its two-hop neighbors as the second anchor, and it uses Algorithm 1 to rank all the nodes it has information on. Node s_0 then selects a one-hop neighbor to repeat this process, until reaching the end of the LWSN.

This iterative solution is detailed in Algorithm 2. This algorithm is executed by some selected nodes in the network, the sink s_0 being the first one (Line 1). At each iteration, the currently selected node s_i builds a topology containing its one and two-hop neighbors (Line 4). Examples of such topologies are presented in Figure 3. Node s_i then defines two nodes in his local topology as anchors (Lines 5 - 12), and it applies the centroid-based ranking algorithm on its local topology (Line 13). We note that, with the exception of the first iteration, node s_i is not one of the two anchors used in Algorithm 1. Running the algorithm on its local topology, s_i updates the ranks of its one-hop neighbors (Line 14) and shares this information with them (Line 15). Finally, after ranking all its one-hop neighbors, s_i selects the one with the highest rank (i.e. the node situated

Algorithm 2 Iterative algorithm

Input: $\mathcal{G} = (\mathcal{S}, \mathcal{L})$, s_0
Output: Sensor ordering

1: $s_i \leftarrow s_0$
2: $stop \leftarrow 0$
3: **while** $stop = 0$ **do**
4: $\quad \mathcal{G}' \leftarrow$ Get-Local-Topology($\mathcal{G}, s_i$)
5: $\quad s_f \leftarrow s_j$, *a ranked node in* $\mathcal{G}'$ *with minimum rank*
6: $\quad$ **if** *There are unranked two-hop neighbors* **then**
7: $\quad\quad s_s \leftarrow$ *Select one as the second anchor*
8: $\quad$ **else**
9: $\quad\quad s_s \leftarrow s_j \in \Gamma(s_i)$ *with minimum one-hop neighbors*
10: $\quad\quad stop \leftarrow 1$
11: $\quad$ **end if**
12: $\quad$ Define-Anchors($\mathcal{G}', s_f, s_s$)
13: $\quad$ Run-Centroid($\mathcal{G}'$)
14: $\quad$ Compute-Nodes-Rank($\Gamma(s_i)$)
15: $\quad$ Transmits $\Gamma(s_i)$
16: $\quad s_i \leftarrow$ Node-With-Max-Coordinate($\Gamma(s_i)$)
17: **end while**

the farthest away) to execute the next iteration (Line 16). We note that, in an iteration, node s_i essentially does processing, sending only one message. This message allows the one-hop neighbors of s_i to update their ranks, and to implicitly select which node will execute the next iteration.

The only question that remains to be answered regarding Algorithm 2 is the choice of the two anchors. At the beginning of the iteration, nodes in the two-hop neighborhood of s_i can be divided in two groups: a group $\mathcal{R}$, containing nodes with ranks that are already defined (although not necessarily correct), and a second group $\mathcal{U}$, for nodes with undefined ranks. For the first anchor, s_i uses the node in $\mathcal{R}$ with the minimum rank, which allows updating the coordinates of all the one-hop neighbors of s_i. At the beginning of the ranking algorithm, $\mathcal{R}$ is empty. In this situation, s_0 is setup as the first anchor. Regarding the second anchor, this node needs to be placed as far as possible from the first one. Therefore, a two-hop neighbor in $\mathcal{U}$ is defined as the second anchor and, in Section VI-B, we propose a metric to guide this choice in order to obtain the best performance. Of course, when the ranking computation approaches the network edge, the choice of a two-hop neighbor might not be possible, and the set $\mathcal{U}$ might even be empty. In this situation, s_i selects, among its one-hop neighbors, the one with the minimum neighborhood size, which is the most likely to be the last node in the network.

Regarding the communication complexity of the proposed solution, we can assume that the communication cost for neighborhood discovery is constant. The overall communication cost for the node ranking part depends on the number of iterations of Algorithm 2, which depends on some network properties like the network diameter or the average node degree. These network properties depend, in turn, on the deployment scenario and environment.

VI. Performance evaluation

So far, we have considered an UDG model, i.e. a network with stable, symmetric and uniform links. However, it is well

known that, in a real WSN deployment, the communication area of a node is never a perfect disk. Our goal in this section is to evaluate the performance of the proposed algorithm under more realistic network conditions.

A. Evaluation methodology

We consider networks with asymmetric links, i.e, between nodes s_i and s_j, we have two independent links $s_i \rightarrow s_j$ and $s_j \rightarrow s_i$. Nodes are linearly and uniformly deployed in an LWSN, with an inter-node distance of 5 m. We define a *missing link* as a link that exists in an UDG scenario, but not in the studied network. To take into account the dynamic nature of links in a WSN, we evaluate the performance of our algorithm when a given number of links are missing. However, to ensure connectivity, we assume that the node deployment guarantees a stable and symmetric link between two consecutive nodes s_i and s_{i+1}. Such a condition can be satisfied during network deployment, by selecting the appropriate hardware or by adapting certain parameters (i.e. transmission power) to the environment. Considering a given network size and a given missing link probability (the same for all the links, except direct physical neighbor, which are always connected), we simulate 2000 different topologies. For each topology, we apply our iterative algorithm, and finally we calculate the ratio of correctly ranked pairs of nodes, as described in Section II.

B. Preliminary observations

The fact the we allow missing links creates new challenges compared with the theoretical UDG scenario. If we consider a simple network with 13 nodes and $\Delta = 6$, in the UDG case the one-hop neighborhood of sink node s_0 would be formed by nodes $s_1 - s_6$. However, this is no longer the case when missing links are allowed. Figure 4 shows, for this toy scenario, the farthest one-hop neighbor of the sink as a function of the missing link probability. This figure shows that, when dynamic conditions are considered, the node coverage area, and implicitly the neighborhood size, are not static as assumed in the UDG model. One direct consequence is that, when the missing link probability increases, the number of iterations required by our node ranking algorithm increases, as less nodes are ranked in each iteration.

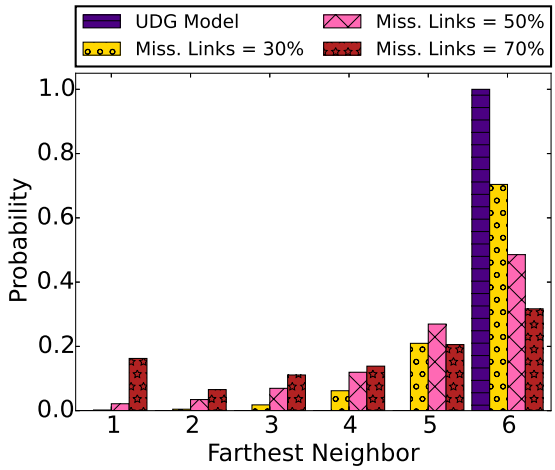

Fig. 4: The coverage area of the sink, presented as the farthest one-hop neighbor, for different missing link probability

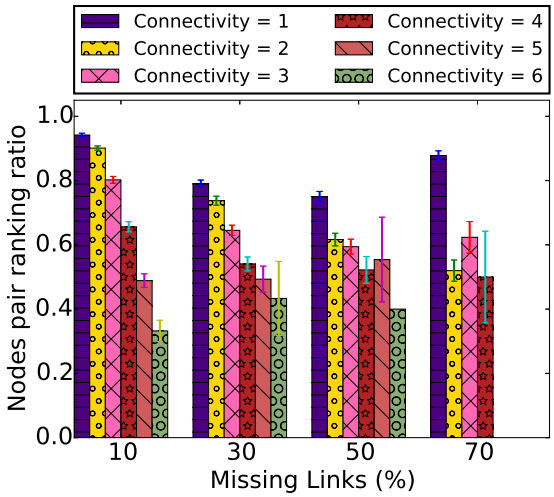

Fig. 5: Correct ranking ratio in the one-hop neighborhood of node s_0, depending on the choice of the second anchor. Results are shown with 95% confidence intervals.

Another consequence of asymmetric and dynamic links can be noticed in the selection of the second anchor for each iteration, as described in Section V-B. As explained, node s_i executing an iteration of the node ranking algorithm should ideally select as a second anchor the most distant unranked two-hop neighbor. However, no information concerning the distance between nodes is available. Moreover, in realistic settings, with asymmetric links, even the definition of the two-hop neighborhood needs to be clarified, as one node might hear another, but not the other way around. Therefore, we design a metric to guide the choice of the second anchor among the two-hop neighbors: $C(s_i, s_j)$, the *connectivity* of nodes s_j related to s_i, defined as the number of nodes receiving messages from s_j and, at the same time, able to send messages to s_i. Formally:

$$C(s_i, s_j) = |\{s_k \in \Gamma(s_i) \backslash s_j \in \Gamma(s_k)\}| \quad (2)$$

Figure 5 presents the ratio of correctly ranked pairs of nodes in the one-hop neighborhood of sink node s_0, depending on the choice of the second anchor in terms of connectivity. This practically corresponds to the first iteration of the node ranking algorithm. The results show that the two-hop neighbor with the lowest connectivity gives the best ranking ratio. We use this insight in the following, to evaluate the overall performance of the iterative ranking algorithm.

C. Algorithm performance

To evaluate the ranking algorithm, we consider a network of 21 nodes, and we vary Δ from 3 to 6. Figure 6 presents the performance of our iterative algorithm as a function of the percentage of missing links and Δ. We note that, since we assume there is always a symmetric link between two physically consecutive nodes, these links are not considered among the possible missing links. This means that, deleting all the other links in the network (i.e. 100% missing links) results in a backbone topology, in which each node has exactly two neighbors: its successor and predecessor in the physical deployment. That is why in such a situation, 100% of pairs of nodes are well ordered, regardless the value of Δ.

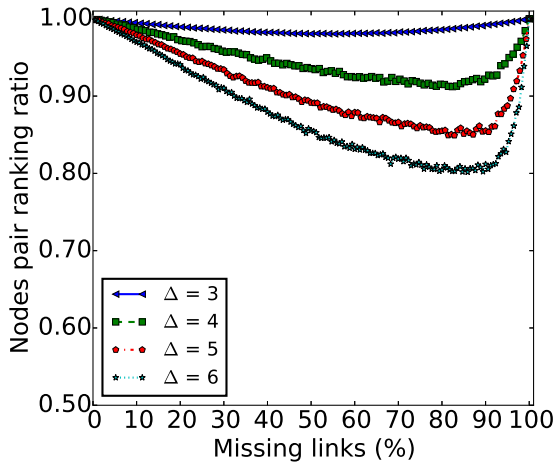

Fig. 6: Ratio of the correctly ranked pairs of nodes as a function of the percentage of missing links, for different values of Δ.

This figure also highlights the good performance of the proposed algorithm: a correct ranking ratio of more than 0.95 is obtained for $\Delta = 3$. We observe that, as the one-hop neighborhood size increases (i.e. higher Δ), the ranking performance decreases. However, in practice, a node with many one-hop neighbors can prune some of them and use in the ranking algorithm only high quality links. The trade-off of reducing the neighborhood size is an increase in the number of iterations required by the ranking algorithm, as it can be noticed in Figure 7. A direct consequence is therefore the increase of the overall energy cost required by the ranking algorithm; we note however that this algorithm is only executed after node deployment or re-deployment. Thus, we argue that it is worth paying the cost induced by link pruning in order to guarantee a good ranking performance.

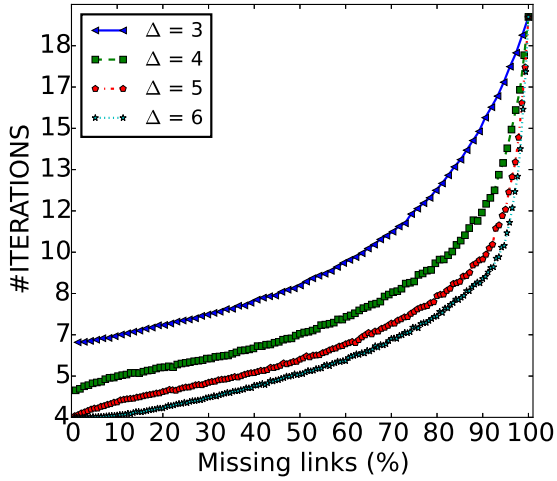

Fig. 7: Total number of iterations as a function of the percentage of missing links, for different values of Δ.

VII. Conclusion

In this paper, we proposed an iterative and distributed centroid-based algorithm for node ranking in a WSN with linear topology. Our algorithm takes as input local information regarding node connectivity and the single manually configured parameter is the first sensor of the network. The proposed algorithm iteratively selects virtual anchors and correctly ranks more and more nodes in the network. At each iteration, all the processing done by the selected node depends on its neighborhood size and not on the number of nodes in the entire network. Thus, our solution is scalable and lightweight considering the network deployment and maintenance. Extensive simulations considering networks with various degrees of connectivity showed the robustness of the approach, as well as demonstrating the high quality of the obtained output sequences. Indeed, even in networks with low connectivity, our solution can produce a sequence with more than 95% of correctly ranked pairs of nodes. The next step of our work is to evaluate the performance of the proposed algorithm in a real deployment.

References

[1] I. Jawhar, N. Mohamed, D.P. Agrawal, "Linear Wireless Sensor Networks: Classification and Applications", *Journal of Network and Computer Applications*, 34(5):1671–1682, Sep. 2011.

[2] F. Stajano, N. Hoult, I. Wassell, P. Bennett, C. Middleton, K. Soga, "Smart Bridges, Smart Tunnels: Transforming Wireless Sensor Networks from Research Prototypes into Robust Engineering Infrastructure", *Ad Hoc Networks*, 8(8):872–888, Nov. 2010.

[3] W. Fisher, T. Camp, V. Krzhizhanovskaya, "Crack Detection in Earth Dam and Levee Passive Seismic Data Using Support Vector Machines", *Proc. ICCS 2016*, San Diego, CA, USA, Jun. 2016.

[4] R. Domga Komguem, R. Stanica, M. Tchuente, F. Valois, "WARIM: Wireless Sensor Networks Architecture for a Reliable Intersection Monitoring", *Proc. IEEE ITSC 2014*, Quingdao, China, Oct. 2014.

[5] Z. Sun, P. Wang, M. Vuran, M. Al-Rodhaan, A. Al-Dhelaan, I. Akyildiz, "BorderSense: Border Patrol through Advances Wireless Sensor Networks", *Ad Hoc Networks*, 9(3):468–477, May 2011.

[6] A. Boukerche, H. Oliveira, E. Nakamura, A. Loureiro, "Localization Systems for Wireless Sensor Networks", *IEEE Wireless Communications*, 14(6):6–12, Dec. 2007.

[7] G. von Zengen, Y. Schroder, S. Rottmann, F. Busching and L.C. Wolf, "No-Cost Distance Estimation using Standard WSN Radios", *Proc. IEEE Infocom 2016*, San Francisco, CA, Apr. 2016.

[8] S. Zhang, X. Liu, J. Wang, J. Cao, G. Min, "Accurate Range-Free Localization for Anisotropic Wireless Sensor Networks", *ACM Transaction on Sensor Networks*, 11(3):51, May 2015.

[9] S. Merkel, S. Mostaghim, H. Schmeck, "Distributed Geometric Distance Estimation in Ad Hoc Networks", *Proc. Adhoc-Now 2012*, Belgrade, Serbia, Jul. 2012.

[10] X. Zhu, G. Chen, "Spatial Ordering Derivation for One-Dimensional Wireless Sensor Networks", *Proc. IEEE ISPA 2011*, Busan, Korea, May 2011.

[11] X. Zhu, X. Wu, G. Chen, "Relative Localization for Wireless Sensor Networks with Linear Topology", *Computer Communications*, 36(1516):1581–1591, Sep. 2013.

[12] Y. Cui, Q. Wang, H. Yuan, X. Song, X. Hu, L. Zhao, "Relative Localization in Wireless Sensor Networks for Measurement of Electric Fields under HVDC Transmission Lines", *Sensors*, 15(2):3540–3564, Feb. 2015.

[13] M.O. Ergin, V. Handziski, A. Behboodi, A. Wolisz, "Determining Node Sequence in a Linear Configuration", *Proc. IPIN 2014*, Busan, Korea, Oct. 2014.

[14] Z. Lotker, M.M. de Albeniz, S. Pérénnes, "Range-Free Ranking in Sensors Networks and Its Applications to Localization", *Proc. Adhoc-Now 2004*, Vancouver, BC, Canada, Jul. 2004.

[15] T. Watteyne, I. Augé-Blum, M. Dohler, S. Ubéda, D. Barthel, "Centroid Virtual Coordinates: A Novel Near-Shortest Path Routing Paradigm", *Computer Networks*, 53(10):1697–1711, Jul. 2009.