Using Adaptive Sensor Ranking to Reduce the Overhead of the Coverage Configuration Protocol

Silvia Santini
ETH Zurich, Institute for Pervasive Computing
Universitätstrasse 6, CH-8092, Zurich, Switzerland
santinis@inf.ethz.ch

***Abstract*—Several application scenarios of wireless sensor networks require coverage of a given region of interest to be guaranteed. In order to establish a coverage preserving configuration, sensor nodes must coordinate and thus invest precious energy resources for both computation and communication. Reducing this overhead is particularly important in applications requiring frequent recomputations of the coverage configuration. In this paper, we focus on these applications and on a particularly efficient coverage configuration protocol, known as CCP. By introducing a novel adaptive sensor ranking technique, we enable CCP to perform a timely selection of the nodes whose activation guarantees for the fulfillment of the coverage requirement. Simulation results show that our approach achieves a 10% reduction of the communication overhead of CCP.**

I. Introduction

Being able to operate unattended for long periods of time, wireless sensor networks (WSNs) enable monitoring environmental phenomena at a fine-grained spatial and temporal scale. Thanks to this characteristic, WSNs are envisioned to be used in countless application scenarios [1]. Irrespectively of the application, however, the actual lifetime of a WSN is limited by the typically scarce and often non-renewable energy resources of its nodes. In particular, since the operation of the radio is known to be the major factor of energy consumption on sensor nodes, limiting its usage is crucial to increase the lifetime of the network. On the other hand, meeting the requirements of WSN applications requires sensor nodes to actively participate in sensing and communication. The efficient operation of a WSN thus requires careful scheduling of nodes participation in sensing and communication.

In typical target detection and tracking applications [2], WSNs are required to provide for area coverage of a given region of interest (RoI). In these scenarios, the network must activate a sufficient number of its nodes so as to guarantee that each point of the RoI is within a given maximal distance from at least one of the active nodes. Nodes selection strategies that allow to fulfill this condition are usually referred to as coverage algorithms [3]. Under the assumption that the number of physically deployed nodes is higher than the number of nodes strictly required to cove the RoI, these algorithms may allow for energy savings by keeping as less nodes active (i.e., available for sensing and communication) as possible [4], [5], [6], [7], [8], [9]. Clearly, the overhead necessary to establish the coverage configuration must be kept low, so as to limit communication and computation and thus save precious energy resources on the nodes. This is particularly important in scenarios requiring the coverage algorithm to be (re-)run several times. For instance, this occurs when coverage must be guaranteed only over a short-lived time interval (e.g., upon detection of an event). Also, coverage algorithms may be used in sensor field reconstruction scenarios. In this case, the required coverage range may change over time and space according to the actual environmental conditions [10, Ch. 4]. Furthermore, a coverage configuration may need to be refreshed regularly for balancing the energy consumption of the nodes and to take possible node failures into account.

In this paper, we focus on the Coverage Configuration Protocol (CCP) [4] and introduce a novel strategy to make it more suitable to be used in scenarios requiring frequent recomputations of the coverage configuration. In particular, we introduce an heuristic sensor ranking method to avoid unnecessary node activations during the execution of the algorithm. Our simulation results show that our ranking strategy allows reducing the communication overhead of CCP by approximately 10%.

The remainder of this paper is organized as follows. We discuss related work and summarize the relevant features and limitations of CCP in sections II and III, respectively. We then introduce our sensor ranking method in section IV and present exeprimental results in section V. Finally, section VI summarizes and concludes the paper.

II. Related Work

In target detection and tracking scenarios, sensor nodes are assumed to be equipped with sensors enabling the detection of a phenomenon (e.g., the presence of a vehicle or person) within a given *sensing range* R_s. Assuming isotropic sensor behavior, the *sensing area* covered by a node n_i can thus be modeled as a discus $D_{R_s}(n_i)$ having the node itself as its center and radius equal to R_s. Guaranteeing constant coverage thus requires scheduling the activations of the nodes so that, at each time instant, each point of the RoI lies within the sensing range of at least one node. More precisely, this type of coverage is known in the literature as *1-coverage*. Generalizing the definition, nodes selection algorithm can guarantee *k-coverage* if, at each time instant, each point of the RoI is within the sensing range of at least k sensors [4].

A central contribution in analyzing the coverage problem in WSNs is offered by Xing et al. in [4]. The authors present theoretical results relating the two concepts of connectivity and coverage. In particular, they show that if the transmission range R_{tx} of the nodes fulfills the condition $2R_s \leq R_{tx}$ and the RoI is 1-covered in the sense we explained above, then 1-connectivity is also guaranteed. The result is also extended to the general case of k-coverage and k-connectivity (a network is said to be k-connected if, to disconnect it, k nodes must be removed). Besides these important theoretical results, the authors also present a coverage algorithm, dubbed CCP. As we also detail in section III, nodes running CCP decide upon their activation by evaluating if their sensing area is already covered by other active nodes. A node does not activate if its sensing area is already covered, while it does become active otherwise. Upon activation, a node communicates its decision to its neighbors through a dedicated broadcast message. Withdrawal from the set of active nodes is also possible.

Several other coverage algorithms work along the same rationale of CCP. For example, Tian et al. [5] propose a different technique to determine whether the sensing area of a node is covered by its active neighbors. Instead, the PEAS algorithm presented in [6] uses an alternative, pull-based approach. To determine whether to become active or not, nodes running PEAS broadcast *probing messages* thereby setting the transmission range R_{tx} equal to a desired probing range R_p. Active nodes receiving the probe broadcast a reply message. The probing node becomes active if it does not receive any reply before a timeout expires. Otherwise, it turns itself off until the next probing round. This approach works well under the assumption of isotropic antenna patterns and adjustable transmission ranges. Unfortunately, both assumptions are hardly met in real WSN deployments.

Several authors also consider the coverage problem in settings in which the active nodes are selected (or deployed) at random over the RoI [7], [8], [9]. For instance, [7] reports the asymptotic conditions necessary to guarantee that a region is (almost always) k-covered by a set of n nodes. The nodes are assumed to have all the same sensing range r and be active or inactive with probability p and $(1-p)$, respectively. The results are derived under the assumption that the RoI is the unit square and the nodes are deployed over it as a uniform grid, uniformly at random or according to a Poisson distribution. In [8] Wan and Yi consider the same problem for both the cases in which the nodes are deployed according to a Poisson or uniform point process. In particular, they show how the probability of coverage changes as the sensing range and number of nodes vary. Furthermore, they note that their results and those reported in [7] are not consistent, possibly due to a different handling of the boundary conditions. For further results we refer the interested reader to [11], [12], [9], and [13].

III. The Coverage Configuration Protocol

Nodes running CCP [4] are assumed to wake up periodically and advertise themselves using HELLO messages. After wake up, a node enters a T_l seconds long LISTEN phase, during which it collects information about the presence, position and state of its neighbors. In particular, neighbors in the ACTIVE state must advertise themselves by sending HELLO messages with high enough frequency. After the LISTEN phase, each node determines whether its sensing area is already covered by neighboring active nodes. A node remains idle or turns itself off if it finds its activation not to be necessary to cover its sensing area. Otherwise, it enters a JOIN phase in which it persists until a timer T_j expires. The node then switches to the ACTIVE state if none of its neighbors advertises itself as active before T_j expires. While in ACTIVE state, nodes continuously collect messages from their neighbors and accordingly reconsider their state. In particular, if a node determines its sensing area to be covered by neighboring active nodes it enters the WITHDRAW state. While in this state, the node waits for a timer T_w to expire before declaring its withdrawal from the active state.

The choice of the values of the join timers T_j significantly influences the performance of CCP and becomes increasingly crucial as the number of executions of the coverage configuration procedure increases. In particular, since nodes communicate their joining of (or withdrawing from) the set of active nodes through a broadcast message, the communication overhead of CCP grows significantly with the number of state changes. Additionally, each time a node receives a JOIN or WITHDRAW message it must perform computation to establish if its sensing area is covered by its active neighbors. If nodes change their state frequently (for instance due to instable links and thus missing notifications) the computational overhead of the protocol rises quickly. Furthermore, if the WITHDRAW notification of a node is lost, part of the network may remain uncovered, since neighbors of that node may still count on its coverage and erroneously decide to withdraw. Thus, limiting the number of withdraws implicitly enhances the reliability of the protocol.

These considerations show that in order to reduce the overhead of CCP it is crucial to minimize the number of nodes becoming active during the initial phase of its execution. To this end, it is particularly important to properly set the values of the join timers T_j and, although less critical, also of the withdrawal timers T_w. In [4] nodes autonomously set these timers as the realizations of random variables uniformly distributed in the intervals $[0, T_j^{max}]$ and $[0, T_w^{max}]$, respectively. In [4] T_j^{max} and T_w^{max}, which clearly represent the maximal values allowed for the join and withdraw timers, are set according to the network density. In particular, in dense networks nodes should be given more time to collect ACTIVE or WITHDRAW messages from their (crowded) neighborhoods and, thus, the values of the T_j^{max} and T_w^{max} timers should be accordingly increased.

Using the above described *random* strategy to set the join and withdraw timers clearly helps preventing collisions of messages sent by nodes simultaneously joining, or withdrawing from, the set of active nodes. However, as suggested in [4], linking the values of the join and withdraw timers to the "utility" of a node for the sensing task would enable a more efficient implementation of CCP. In particular, the authors of [4] hint at the fact that nodes covering more uncovered area should be assigned shorter join timers. However, the definition of proper heuristics to rank the relevance of a sensor has not been further investigated. In the following section we introduce a set of strategies that can provide for this ranking. In section V, we then show how our approach allows reducing CCP's communication overhead by approximately 10%.

IV. Adaptive Sensor Ranking

Let us assume each node n_i of a WSN can be assigned a scalar value $\Psi_i \in [0,1]$ that describes its utility for a given sensing task. In particular, let Ψ_i represent an estimate of the probability with which the node n_i is required to become active in order to guarantee coverage of its sensing area. The corresponding join timer of node n_i can then be determined as:

$$T_j(i) = T_j^{max}(1 - \Psi_i). \tag{1}$$

Indeed, a node having utility, or *sensor rank*, Ψ_i equal to 1 will be required to become active irrespectively of the decisions taken from its neighbors. Therefore, it should communicate its activation as quickly as possible so as to enable other nodes to refrain from becoming active. On the other hand, an early activation of low-utility nodes may lead to a high number of withdrawals, since their sensing area may result covered by high-utility nodes. Thus, the lower the rank Ψ of a node, the higher should be the value of its join timer T_j, as expressed by equation 1.

The question we would like to answer in the remainder of this section is thus how to compute good estimates of the rank Ψ of the nodes of a WSN. In related application fields, the relevance of a node for a sensing task can often be approximated using the area of its Voronoi cell [14]. However, distributed computation of Voronoi cells in WSNs requires knowledge of the nodes' neighborhoods over several hops and has an overall high messaging and computational cost [15]. On the other hand, a centralized computation has well-known drawbacks in terms of scalability. Instead, our goal is to offer a good estimate of the rank of a node while enabling its computation in a distributed fashion and using information that is anyway collected during the execution of CCP.

A. Ranking Based On Local Density

A rough estimate of the sensor rank of a node can be derived from the cardinality of its neighborhood. A node in a scarcely populated neighborhood has indeed a higher probability to be required to become active with respect to nodes with a large number of neighbors. Assuming that two nodes n_i and n_j have both equal sensing range R_s and Euclidean distance d_{ij} to each other, we define the *sensing neighborhood* of a node n_i to be the set of nodes $\mathcal{N}_s(n_i, R_s) = \{n_j : d_{ij} \leq R_s\}$. Accordingly, we can define the rank of a node n_i as:

$$\Psi_i = \frac{1}{|\mathcal{N}_s(n_i, R_s)|}, \tag{2}$$

where $|\mathcal{N}_s(n_i, R_s)|$ indicates the number of elements of the set $\mathcal{N}_s(n_i, R_s)$, i.e. its cardinality. To determine its sensing neighborhood, a node n_i must clearly be aware of the presence of nearby nodes, as well as know their distance. Coverage protocols in general, and CCP in particular, typically require nodes to know about the existence and position of their neighbors in order to compute the coverage configuration. We can therefore assume this information to be available without any loss of generality. In practical settings, however, nodes may only be aware of the presence of neighbors to which they can establish a direct communication link. Although this may induce errors in the determination of the cardinality of the neighborhoods, these errors have a negligible influence on the effectiveness of our ranking strategies, especially as the density of the network increases.

Since setting the join timers using equations 1 and 2 makes them proportional to the density of the sensing neighborhoods of the nodes, we refer to this method as the *density (S)* strategy. This method is particularly straightforward to implement and, with respect to CCP's default *random* strategy, it allows for some gains in term of communication, as we show in section V. However, the *density (S)* strategy only takes into account the number of nodes present in a node's neighborhood but ignores their actual spatial distribution. This causes nodes having the same number of neighbors to have the same rank, although their neighborhoods may have completely different characteristics. For instance, a node having its neighbors uniformly distributed around it will be assigned the same rank of a node having all its neighbors localized within a small spatial sector.

B. Ranking Based On Weighted Local Density

To amend for the above mentioned drawback of the *density (S)* strategy we propose a slightly more sophisticated ranking method. In particular, we suggest computing the rank of a node n_i considering the distance between n_i and the nodes in its sensing neighborhood. Intuitively, the absence of close-by neighbors signals that a node has a high probability to be required to become active and this should thus make its rank increase. Instead, the presence of close-by neighbors should make the rank decrease. To capture the geometry of the neighborhood of a node n_i, we thus suggest to compute the rank Ψ_i of a node n_i as the weighted density of n_i's sensing neighborhood. To this end, we must assign a weight ϕ_{ij} to each $n_j \in \mathcal{N}_s(n_i, R_s)$. To compute these weights, we resort to inverse distance weighting (IDW).

IDW is a technique often used in field reconstruction algorithms to determine the appropriate weights of individual samples [16], [17]. In this context, the weights can be computed using an arbitrary function ϕ and an appropriate normalization, as discussed in [16]. There exist several different functions that can provide for a proper IDW metric [16], but we consider here a simple linear weighting and set:

$$\phi_{ij} = 1 - \frac{d_{ij}}{R_s}, \tag{3}$$

where d_{ij} represents the Euclidean distance between nodes n_i and n_j. Using the weights ϕ_{ij} we can include information about the relative distance between a node n_i and its neighbors in the computation of the rank Ψ_i. However, this computation can be additionally refined by taking into account also information about the position of the neighbors with respect to n_i. To this end, we virtually divide the neighbors of a node n_i into N_{sets} sets $S_{ik}, k = 1, .., N_{sets}$. If the network is deployed on a line (1-dimensional case), the node n_i can assign each neighbor n_j to its "left" ($|s_i| >= s_j$) or "right" ($(|s_i| < s_j$) neighborhood, thus $N_{sets} = 2$. If the nodes are deployed on a plane (2-dimensional case), the sets may correspond to an arbitrary number of N_{sets} disjoint circular sectors spanning the circle centered on the node and having radius R_s. The contribution to the rank of node n_i of the neighbors in the kth sector is then given by:

$$\Psi_{ik} = \frac{1}{1 + \sum_{j=1}^{|S_{ik}|} \phi_{ij}}, \tag{4}$$

For simplicity, we rewrite equation 4 as:

$$\Psi_{ik} = \frac{1}{\sum_{j=0}^{N_{ik}} \phi_{ij}} \tag{5}$$

where the "neighbor" of index $j = 0$ corresponds to the node n_i itself and, thus, $d_{i0} = 0$. The rank Ψ_{ik} of sector k clearly represents the weighted local density of node n_i relative to sector k and provides an estimate of the probability with which node n_i is required to be active in order to cover the area spanned by sector k. An appropriate aggregate (e.g., minimum or average) of the ranks of all N_{sets} sectors then gives the desired rank Ψ_i of the node n_i. In the context of this work, we use the average as aggregation function. Accordingly, we set:

$$\Psi_i = \frac{1}{N_{sets}} \sum_{k=1}^{N_{sets}} \Psi_{ik}. \tag{6}$$

If node n_i has an empty sensing neighborhood, Ψ_i takes its maximal value 1. Instead, Ψ_i asymptotically goes to zero as the number of nodes in n_i's neighborhood increases. This definition of the rank Ψ_i can well approximate, as desired, the probability with which a node n_i must become active during the execution of CCP. Indeed, if n_i has an empty sensing neighborhood, it is forced to become active and, thus, its probability of activation is 1. In the other extreme case in which all nodes in n_i's sensing neighborhood occupy the same physical spot (and, thus, $d_{ij} = 0, \forall n_j \in \mathcal{N}_s(n_i, R_s)$), its probability of activation will simply be $1/|\mathcal{N}_s(n_i, R_s)|$. In other words, n_i should ideally become active once every $|\mathcal{N}_s(n_i, R_s)|$ runs of CCP since its shares the "sensing responsibility" for its sensing area with its $|\mathcal{N}_s(n_i, R_s)|$ "identical" peers. In all intermediate cases the presence of a neighbor n_j at a certain distance d_{ij} will "relieve" the node n_i of an amount of its sensing responsibility that is inversely proportional to the distance d_{ij}.

We refer to a strategy that sets the join timers of the nodes using equations 1 and 6 as the *idw* strategy. In section V we show that this method allows for performance improvements with respect to the *density (S)* strategy, although it induces only a little additional computational cost and memory overhead.

In static networks, however, both the *idw* and the *density (S)* strategies have an important drawback. If nodes do not change their position or do so only infrequently, the sensor ranking given by these methods will not change across different runs of CCP. This means that, up to little differences due to the vagaries of wireless communication, also the set of nodes becoming active at each run will not change. Clearly, this hampers the possibility of balancing the overall energy consumption of the nodes. To cope with this problem, we resort to randomization, as explained below.

C. Load Balancing

To mitigate the above mentioned load balancing inefficiency of the *idw* strategy, we suggest to use the value of Ψ_i as a probabilistic ranking of the node. In other words, the value of the timer $T_j(i)$ of a node n_i can be set as:

$$T_j(i) = T_j^{max} \cdot p(\Psi_i), \tag{7}$$

where $p(\Psi_i)$ is a value drawn at random from a uniform distribution between 0 and $1 - \Psi_i$. In the following section we show that this strategy, dubbed *idw random*, not only allows for better load balancing but also provides for a further reduction of the communication overhead of CCP. Nonetheless, the *idw random* strategy still makes high-rank nodes become active more frequently than low-rank nodes. Therefore, high-rank nodes will likely drain their batteries sooner than others and, thus, the load balancing problem discussed above, although mitigated, still persists. As discussed in [18] and [10, Sect. 4.8], an effective way to preserve the energy of high-rank nodes consists in limiting their participation in other network activities, like, e.g., routing.

V. Evaluation

We implemented CCP on the well-known Matlab simulation environment and studied its performance using different methods to determine the values of the timers T_j. In particular, we consider the heuristics *random* (proposed in [4]), as well as *density (S)*, *idw*, and *idw-random*, introduced in the previous section. As in [4], we assume the value of T_j^{max} to be fixed a priori and available to all nodes. For the sake of completeness, we also consider an additional ranking strategy, dubbed *density (C)*, which works along the same rationale of the *density (S)* but considers the cardinality of the communication neighborhood instead of that of the sensing neighborhood. In the context of this work, we define the *communication neighborhood* of a node n_i to be the set of nodes $\mathcal{N}_c(n_i, R_{tx}) = \{n_j : d_{ij} \leq R_{tx}\}$, where R_{tx} indicates the transmissions range of the nodes' on-board radio transceiver. Since we include this ranking strategy only for the sake of comparison, we do not consider more complex definitions of the communication neighborhood, which would need taking into account anisotropic antenna propagation patterns and the vagaries of the wireless channel.

In our simulation setup, N_{tot} sensor nodes deployed at random over a square region collect information about their neighborhood for a time interval T_l (*listen phase*). The initial value of T_l is set proportional to the average density of the network, as suggested in [4]. After T_l expires, every node n_i holds an updated list of its neighbors, along with their positions. Then, the network enters the *activation phase* in which each node determines whether it must become active or not. The order in which nodes decide upon their activation depends on the values of the timers T_j and is therefore determined by the strategy used to set these values. After the timer T_j^{max} expires, all nodes are assumed to have decided upon their activation and the network can thus enter the *withdrawal phase*. The order in which nodes decide upon their withdrawal is the same order used in the activation phase. This can be easily implemented by making nodes start a withdraw timer $T_w = T_j^{max}$ immediately after they make the decision to become active. The total duration of the listen, activation and withdrawal phases is, therefore, $T_{tot} = T_l + 2T_j^{max}$. During this time-frame sensor nodes must keep their radio circuitry powered on to receive and send status messages. Nodes that do not become active during the activation phase can immediately power off their radios, unless they are required to remain active for routing purposes. On this regard we should also recall that, if the communication range is at least twice as big as the sensing range, connectivity can be guaranteed by the set of active nodes [4]. In this case, inactive nodes can switch off their radios as soon as their activation timer expires without further restrictions.

To quantify the communication overhead of CCP, let C_{tx} and C_{rx} represents the cost of transmitting and receiving a packet, respectively. If $\mathcal{N}_c(n_i, R_{tx})$ is the communication neighborhood of node n_i, the communication cost of the activation of node n_i is $C_{Ai} = C_{tx} + |\mathcal{N}_c(n_i, R_{tx})| \cdot C_{rx}$. If S_{ANbw} is the set of nodes in ACTIVE state before the withdrawal phase starts, then the total communication cost of the activation phase is: $C_A = \sum_{j \in S_{ANbw}} (C_{tx} + |\mathcal{N}_c(n_j, R_{tx})| \cdot C_{rx}) = |S_{ANbw}| C_{tx} + C_{rx} \sum_{j \in S_{ANbw}} |\mathcal{N}_c(n_j, R_{tx})|$. Similarly, if S_{WN} is the set of nodes that changes its state during the withdrawal phase, the total cost of this phase in terms of communication is: $C_W = \sum_{j \in S_{WN}} (C_{tx} + |\mathcal{N}_c(n_j, R_{tx})| \cdot C_{rx}) = |S_{WN}| C_{tx} + C_{rx} \sum_{j \in S_{WN}} |\mathcal{N}_c(n_j, R_{tx})|$. The total communication overhead of CCP can thus be computed as:

$$C_{CCP} = C_A + C_W. \tag{8}$$

In equation 8 we neglect the cost of the listen phase, which is independent of the strategy used to set the activation timers. The costs C_{tx} and C_{rx} can be set using, for instance, the method proposed in [19]. However, since we are concerned only with the number of packets that are transmitted and received, we can simply assign a unitary value to both C_{tx} and C_{rx}. The assumption that $C_{tx} = C_{rx}$ holds for several sensor network prototyping platforms (e.g., Tmote Sky).

During the activation and withdrawal phase, sensor nodes must keep their radios in idle listening, since they cannot predict the time instants at which neighbors will possibly send their activation or withdraw beacons. This additional source of energy consumption is not included in equation 8 since it is common to all the strategies considered to set the join timers. However, reducing the number of nodes becoming active in the activation phase allows to switch off more nodes earlier, and, thus, to reduce the amount of time these nodes must persist in idle listening. Further, we do not consider message losses since they equally affect the considered sensor ranking strategies.

For our simulation study we run CCP over 25 different random network configurations. For ranking strategies involving elements of randomness (namely the *random* and *idw-random* methods), we run 25 trials for each configuration. In our experiments, we consider a variable number of nodes N_{tot}, from 200 to 300 deployed over a square region of sides $L_x = L_y = 100m$. The transmission range R_{tx} has been fixed to $25m$ while the value of R_s increases from $0.375R_{tx}$ to $0.5R_{tx}$. For each experiment, we compute the percentage of active nodes before and after the withdrawal phase, as well as the total communication overhead as specified by equation 8. Since the communication overhead of the different strategies depends upon the local sensing density of the nodes, we display these comprehensive results as the ratio $\pi R_s^2 N_{tot} / L_x L_y$ increases.

Figure 1 shows that the number of nodes required to be active after the withdrawal phase is almost identical for all the considered strategies. This shows that, in terms of number of nodes that eventually remain active, the considered ranking strategies are interchangeable. Therefore, a higher number of active nodes before the withdrawal phase constitutes a net cost in terms of communication overhead, since it does not allow for any performance gain afterwards.

To quantify this cost, figure 2 displays the total number of messages sent and received during the execution of CCP as a function of the average density of the network increases. Since we are evaluating the achievable improvements with respect to the *random* strategy proposed in [4], we normalize the results of each experiment to the cost of this strategy. Thus, figure 2 shows the achievable gains, or losses, in terms of communication overhead that incur in using a different strategy rather than the *random*. In particular, it shows that *idw* and *density (S)* metrics perform well for low densities. As the average number of nodes in a sensing neighborhood increases, however, randomized strategies outperform deterministic methods. Also, we can observe that the *idw-random* strategy requires about 10% less communication with respect to the plain *random* assignment method. These results thus show that our heuristics to set the activation timers T_j are effective in reducing the communication overhead of CCP.

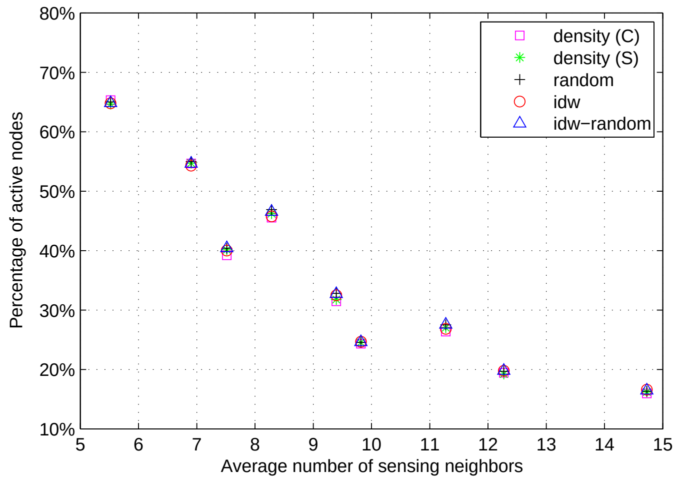

Figure 1. Number of active nodes after the withdrawal phase as the average number of sensing neighbors increases. Results are in percentage with respect to the the total number of available nodes N_{tot}. Experimental setting (2-dimensional case): $L_x = 100m$, $L_y = 100m$, $R_{tx} = 25m$, $2R_s = 18.75, 21.875$, and $25m$, $N_{tot} = 200, 250$, and 300.

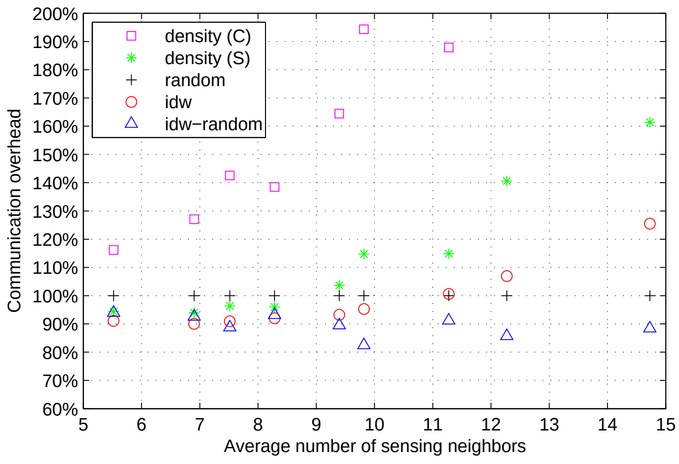

Figure 2. Communication overhead of CCP using different strategies to set the join timers of the nodes as the average number of sensing neighbors increases. Results are in percentage with respect to the overhead of the *random* strategy. Experimental setting (2-dimensional case): $L_x = 100m$, $L_y = 100m$, $R_{tx} = 25m$, $2R_s = 18.75, 21.875$, and $25m$, $N_{tot} = 200, 250$, and 300.

VI. Conclusions

In this paper, we focused on the coverage configuration protocol by Xing et al. [4] and investigated possible strategies to reduce its communication overhead. In particular, we introduced a set of heuristics that leverages available local information to set the timers that regulate CCP's execution. Our experimental results show that our adaptive strategy is effective in reducing the number of communications CCP requires to establish a coverage configuration.

References

[1] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless Sensor Networks: A survey," *Computer Networks*, vol. 38, no. 4, pp. 393–422, March 2002.

[2] S. Pattem, S. Poduri, and B. Krishnamachari, "Energy-quality tradeoffs for target tracking in wireless sensor networks," in *Proceedings of the 2nd International Conference on Information Processing in Sensor Networks (IPSN 2003)*," Palo Alto, CA, USA, 2003, pp. 32–46.

[3] M. Cardei and J. Wu, "Energy-efficient coverage problems in wireless ad-hoc sensor networks," *Computer Communications*, vol. 29, no. 4, pp. 413 – 420, 2006, current areas of interest in wireless sensor networks designs.

[4] G. Xing, X. Wang, Y. Zhang, C. Lu, R. Pless, and C. Gill, "Integrated Coverage and Connectivity Configuration for Energy Conservation in Sensor Networks," *ACM Transactions on Sensor Networks (TOSN)*, vol. 1, no. 1, pp. 36–72, August 2005.

[5] D. Tian and N. D. Georganas, "A Coverage-Preserving Node Scheduling Scheme for Large Wireless Sensor Networks," in *Proceedings of the 1st ACM International Workshop on Wireless Sensor Networks and Applications*," Atlanta, GA, USA, September 2002, pp. 32–41.

[6] F. Ye, G. Zhong, J. Cheng, S. Lu, and L. Zhang, "PEAS: A Robust Energy Conserving Protocol for Long-Lived Sensor Networks," in *Proceedings of the 23rd International Conference on Distributed Computing Systems (ICDCS 2003)*," Providence, RI, USA, May 2003, pp. 28–37.

[7] S. Kumar, T. H. Laiand, and J. Balogh, "On k-Coverage in a Mostly Sleeping Sensor Network," *Wireless Networks*, vol. 14, no. 3, pp. 277–294, June 2008.

[8] P.-J. Wan and C.-W. Yi, "Coverage by Randomly Deployed Wireless Sensor Networks," *IEEE/ACM Transactions on Networking (TON), Special Issue on Networking and Information Theory*, vol. 14, pp. 2658–2669, June 2006.

[9] W. Choi and S. K. Das, "Coverage-Adaptive Random Sensor Scheduling for Application-Aware Data Gathering in Wireless Sensor Networks," *Elsevier Computer Communications*, vol. 29, no. 17, pp. 3467–3482, November 2006.

[10] S. Santini, "Adaptive sensor selection algorithms for wireless sensor networks," Ph.D. dissertation, ETH Zurich, Zurich, Switzerland, October 2009.

[11] F. Garwood, "The Variance of the Overlap of Geometrical Figures with Reference to a Bombing Problem," *Biometrika*, vol. 34, no. 1/2, pp. 1–17, January 1947.

[12] P. Hall, *Introduction to the Theory of Coverage Processes*, ser. Wiley Series in Probability and Mathematical Statistics. New York Chichester Brisbane Toronto Singapore: John Wiley & Sons, October 1988.

[13] H. Zhang and J. C. Hou, "On deriving the upper bound of α-lifetime for large sensor networks," in *Proceedings of the 5th ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc 2004)*," Tokyo, Japan, May 2004, pp. 121–132.

[14] F. Aurenhammer, "Voronoi diagrams – A Survey of a Fundamental Geometric Data Structure," *ACM Computing Surveys*, vol. 23, no. 3, pp. 345–405, September 1991.

[15] B. A. Bash and P. J. Desnoyers, "Exact Distributed Voronoi Cell Computation in Sensor Networks," in *Proceedings of the 6th International Conference on Information Processing in Sensor Networks (IPSN 2007)*," Cambridge, MA, USA, April 2007, pp. 236–243.

[16] D. Shepard, "A Two-Dimensional Interpolation Function for Irregularly-Spaced Data," in *Proceedings of the 23rd ACM National Conference/Annual Meeting*, August 1968, pp. 517–524.

[17] F. Marvasti, Ed., *Nonuniform Sampling: Theory and Practice*, ser. Information Technology: Transmission, Processing and Storage. Berlin / Heidelberg: Springer, 2001.

[18] M. Perillo and W. R. Heinzelman, "An Integrated Approach to Sensor Role Selection," *IEEE Transactions on Mobile Computing*, vol. 8, no. 5, pp. 709–720, May 2009.

[19] W. R. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "An Application-Specific Protocol Architecture for Wireless Microsensor Networks," *IEEE Transactions on Wireless communications*, vol. 1, no. 4, pp. 660–670, October 2002.