Sink Placement without Location Information in Large-Scale Wireless Sensor Networks

Wint Yi Poe
disco | Distributed Computer Systems Lab
University of Kaiserslautern, Germany
poe@informatik.uni-kl.de

Jens B. Schmitt
disco | Distributed Computer Systems Lab
University of Kaiserslautern, Germany
jschmitt@informatik.uni-kl.de

ABSTRACT

Wireless Sensor Networks (WSN) comprise a large number of sensor nodes that possess scarce energy supplies. To minimize energy consumption and to consequently extend the lifetime of WSNs, we propose a local search technique for sink placement in WSNs. In addition, a proper sink placement plays a vital role in performance-sensitive WSN applications. Since it is not feasible for a sink to use global information, especially for large-scale WSNs, we introduce a self-organized sink placement (SOSP) strategy that combines the advantages of our previous works [7] and [8]. Besides, this paper is a substantial extension of [9]. The goal of this research is to provide a better sink placement strategy with a lower communication overhead. Avoiding the costly design of using the nodes' location information, each sink sets up its own group by communicating to its n-hop distance neighbors. While keeping the locally optimal placement, SOSP exhibits a better solution quality with respect to communication overhead and computational effort than previous solutions. To analyze performance issues, especially the worst-case delay of a given WSN, we use a methodology called sensor network calculus [10].

Categories and Subject Descriptors

C.2 [**Computer-Communication Networks**]: Network Architecture and Design—*Wireless Communication*; C.4 [**Performance of Systems**]: Design studies

General Terms

Algorithm, Design, Performance

Keywords

Sink Placement, Worst-Case Delay, Network Calculus, Wireless Sensor Networks.

1. INTRODUCTION

Research activity in the area of WSNs has grown dramatically in the past few years and was motivated by a vast array of potential applications. Unlike traditional networks, WSNs may consist of several thousands of sensor nodes for unattended operations. Consequently, various design and control techniques used in traditional networks cannot be applied directly in WSNs. Furthermore, the sensors have the ability to communicate with each other either by sending their own data or by routing the others' data via multi-hop communication to a base station, often called a sink. As a result, the longer the hop distances from a node to the sink, the greater the delay and energy consumption will be. Thus, the lifetime of the WSN becomes shorter. Energy consumption is often assumed to be the most critical issue in WSNs due to their battery operation, which probably constitutes one of the main differences from traditional networks. Thus, all design and control processes are usually focused on minimizing energy consumption as much as possible.

Typical scenarios for large-scale WSNs contain multiple sources and multiple sinks. Compared to a single sink, multiple sinks provide better manageability of WSNs. Depending on different criteria, network designer has to plan a WSN, which not only optimizes energy consumption but also achieves a high system performance. Performance issues in WSNs play a vital role in many applications. In time-sensitive WSN applications, the maximum allowable message transfer delay must be bounded. Hence, it is crucial to develop an algorithm to minimize the maximum worst-case delay in WSNs. Avoiding the costly design of using the nodes' locations information, we introduce a self-organized sink placement (SOSP) algorithm with lower computation and communication overhead to minimize the maximum worst-case delay.

In WSNs, sinks can be either mobile or stationary. For a mobile sink, the dynamic nature of routing becomes the most challenging issue, although a lot of benefits from this mobility also arise. Using the advantages of mobility, we consider mobile sinks for the initial selection of the best sink placement, and later use stationary sinks to collect data packets from the sensor nodes via multi-hop communication.

In case of the improper deployment of sensor nodes and sinks, it is difficult to control a running network and to that end energy consumption and system performance are negatively affected. For a random-based deployment process of a WSN (consider an airplane to distribute the nodes), the sink placement becomes an important criterion for the network designer to increase the network lifetime and system performance. In a WSN, multiple sinks in proper locations can strongly decrease the amount of energy use and the message transfer delay in communication, mainly as the effect of a shorter multi-hop distance between sensor nodes and sinks. For that reason, we focus on a multiple sink location (or placement) problem in self-organizing manner in order to suit the requirements of large-scale WSNs.

The rest of the paper is organized as follows: In Section 2 we discuss the sink location problem and its related work. After that we introduce the SOSP algorithm alongside with deployment, grouping, sink placement selection, and operational phases in Section 3. Section 4 contains a detailed performance evaluation of SOSP against previously proposed benchmarks and near optimal sink placement strategies. Accordingly, we briefly introduce these strategies for comparison of the worst-case analysis method. Finally, we conclude the paper in Section 5.

2. RELATED WORK

In this section, we discuss the sink location problem and its relation to the previous work in WSNs. In WSNs, data packets traverse from the sensor nodes to a sink through multi-hop communication owing to expensiveness or even infeasibility of direct communication. Therefore, a sensor node is not only a source but can also be a router. For large-scale WSNs, if a sensor node acts as a router close to a sink, it can experience quite high amounts of data from other nodes flowing through it. As a result, the traffic intensity in the network becomes high and data transfer may be considerably delayed. What is more, if the sink is located far apart from some nodes, their hop distances increase, simutaneously amplifying the message transfer delay. Therefore, we assume that the path with the lowest hop count is taken in order to minimize the maximum worst-case delay.

The general sink location problem is NP-complete, so finding the exact optimal sink placement is very hard. Some well-known approaches for finding the optimal solution include integer linear programming, exhaustive search and iterative clustering. We also developed a heuristic method [8], which performs well and delivers near-optimal solution. Integer linear programming and exhaustive search only work for small-scale WSNs. They use the global knowledge of the network such as the nodes locations, thus position-awareness of the sensor nodes becomes a critical issue. The sensor nodes equipped with global positioning system (GPS) hardware are for many applications too expensive. Of course, there are plenty localization algorithms that can work quite well under some a priori knowledge of a few nodes' positions and provide an approximation of others. Nonetheless, even this might often be inconvenient to assume in large-scale WSNs due to the related communication overhead. That is why, in this paper, we make no assumptions on a priori global knowledge, apart from minimum knowledge about the size of the sensor field and based on this develop the SOSP algorithm.

Next, we discuss two main categories of sink placement strategies in WSNs: (1) using the global knowledge of the sensor nodes' locations and (2) based on the estimates of the nodes' locations. In the first group, the node locations may be obtained from GPS receivers or are simply known from a planned deployment process. Obviously, it suits only small-scale WSNs because of the corresponding computation and communication overhead. The second category is sink placement based on the estimation of node locations, which uses, for example, anchor points. Yet, these approaches still seem infeasible for large-scale WSNs.

Although sink placement is an obvious problem to be solved in WSN design, in literature, it has been addressed surprisingly seldom compared to other areas, as for example, routing and localization in WSNs. Depending on various design criteria, a number of papers, for example, [6] and [4] address different sink placement methods with different intentions for the main purpose of network lifetime extension. In most cases, clustering is considered as a common way of grouping. There are many good clustering algorithms in literature, which are mainly divided into hierarchical and non-hierarchical methods [17]. If a clustering method is applied in WSNs, the cluster head or the centroid of a cluster is usually considered as a sink. Yet, clustering algorithms normally require global knowledge, such as where the nodes are located, what the energy levels of the nodes are, and so forth. This drives conventional clustering approaches to become infeasible for large-scale WSNs. Another somewhat related work can be found in [16], where the problem of optimally steering mobile sinks is approached by using an electrostatic model for maximizing the lifetime of a WSN. A similar approach is discussed in [2], which also deals with effective control of mobile sinks.

A related work on which we build in this research is our previous research on sensor network calculus (SNC) [10, 14, 15]. Based on network calculus [5], SNC was proposed and customized as a framework for the worst-case analysis in WSNs. In our research, we use the DISCO network calculator [11], which is an open source toolbox for the worst-case analysis written in Java^{TM}. All network calculus operations and different network analysis algorithms can easily be used from this library: for example, end-to-end service curve calculation, total flow analysis, separated flow analysis, PMOO analysis, etc. Using the foundation of sensor network calculus, we calculate the worst-case end-to-end delays for each flow and find the maximum worst-case delay in the field.

3. SELF-ORGANIZED SINK PLACEMENT

In this section we present the algorithm of SOSP in detail. In SOSP, the only assumption on a priori knowledge we make is that sensor nodes are deployed over a circular field shape. Note that one can always draw a circle around a differently shaped field. A generalization from this assumption should be straightforward. Such a lax assumption is key to applicability in large-scale WSNs.

3.1 Algorithm Overview

After the (random) node deployment phase, the algorithm chooses the initial sink placements at the the Center of Gravity of the Sectors of the Circle (CGSC), which results from an equal sectorization with the number of sinks. The mathematical expression for the CGSC is given in Section 4. Next, the sinks, now located at CGSCs, perform grouping, starting from a 1-hop neighbors set in a wave to an n-hop neighbors set. The value of n varies according to the number of sensor nodes, sinks, node density, and transmission ranges. In our algorithm, we assume that sensor nodes are homogeneous and that both sensors and sinks use the same transmission range with a given node density. The simulation results of the n-hop values having different sensor nodes and sinks are discussed in more detail in the subsection on the grouping phase. Note that the maximum value of n must be considered to obtain a fully connected network. Furthermore, we estimate the locations of the 1-hop neighbors which are required for selecting sink candidate locations. Without the supporting GPS hardware, we estimate the locations by using trilateration with the time of arrival (TOA) method [3]. In TOA, the distances to the nodes are estimated time spent by the signal transmitted from the sink. Time spent in transit is converted into the distance travelled to calculate the distance between a sink and a node. Note that this method is better suited for outdoor WSNs for signal interference with obstacles that causes lower accuracy. Knowing the estimated distance, we can calculate the location of the node, which is demonstrated in the subsection on the sink location selection phase. In SOSP, we need to estimate locations only for the 1-hop neighbors sets. From this location information, we define a set of fixed candidate locations for sink placement inside the search space of the 1-hop neighbors region. Finally, the best candidate is selected for sink placement. Here, we assume that a mobile sink performs the tasks of determining the 1-hop neighbor nodes' locations (by doing the TOA measurements from at least three different positions) and the computation of the best sink placement for each group. In the following subsections, we discuss four operational phases of the SOSP algorithm in detail.

3.2 Deployment

During the design phase of WSNs, the designer knows only the number of sensor nodes and sinks to be deployed. Both sinks deployment and nodes deployment affect the performance of the WSN. The nodes deployment can be either planned or random. In a planned node deployment, sensor nodes are manually placed so that their locations are known, whereas for a random deployment, sensor nodes are positioned at locations which are not known with certainty. Planned node deployments are usually only suitable for small-scale WSNs for cost issues. Due to our interest in large-scale WSNs, in this paper we focus on random node deployments. To illustrate where a random node deployment would be inevitable, let us assume that sensor nodes are distributed by an airplaine, for example, in an environmental monitoring application.

3.3 Grouping

Grouping is an important issue for large-scale WSNs for manageability by keeping operations local. Although conventional clustering is considered as a usual way of grouping, we focus on self-organized grouping, which improves in terms of computation and communication overhead. In order to build a group, it is necessary to have an inital location which should be a good starting point for sink placement. The sensor field shape is assumed to be circular in our experiment. Since typical large-scale WSNs are designed with multiple sinks, it is possible to create groups whose number corresponds to the number of sinks. Based on this, we initially choose the sink placement at each CGSC. To minimize the maximum worst-case delay a sink placement at CGSC produces a fair result as presented in [7]. So, we first place at CGSCs mobile sinks that connect to their 1-hop neighbor nodes by broadcasting a message. (Any node here that can hear a message from the sink is defined as a 1-hop neighbor, also referred to as a 1-hop distance neighbor.) Upon receiving this message, the nodes within the sink's transmission range communicate back to their sink (we use a random send time to avoid collisions for these replies). This self-organized grouping of neighboring nodes continues to the next hops up to the set of n-hop neighbors, in which n is set according to simulative results (provided in Subsection 3.3.2). A priori knowledge of the n-hop value helps to minimize the energy consumption while keeping the balance of the group sizes. An example of grouping in a 50-node network with 3 sinks is illustrated in Fig. 1.

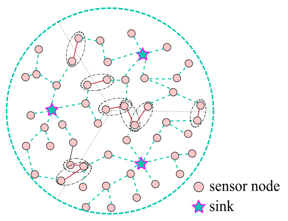

Figure 1: Grouping for 50-node network with 3-sink.

In our algorithm, a node is not allowed to communicate with more than one sink to avoid the nodes' duplication and to maintain a fully connected network, as shown in Fig. 1. In the grouping phase, if a node is already in a group it is then owned by that group, although it may have a link to another group (perhaps, it may even have a lower hop distance to the other sink). For those nodes that act as gateways to other groups, we have the chance to choose the nearest sink (i.e., the one with shorter hop distance) during the operation phase. Such gateway nodes are marked inside an oval in Fig. 1. What is more, this characteristic is very important for fault-tolerance because a node may take an alternative route to another sink in case of a node failure inside the local group.

In addition, the determination of the 1-hop neighbors' locations is necessary for the selection of the best sink placement during the grouping phase. In the following subsections, we describe (1) the determination of 1-hop neighbors' locations from estimated distances and (2) the simulation results of the maximum n-hop distances with respect to related factors, such as the number of sensor nodes, sinks, transmission range, and node density.

3.3.1 Determination of Distances and Locations

While grouping the 1-hop neighbor nodes, the algorithm first calculates their approximate locations. From three main approaches to localization: Proximity-based approach, triangulation and trilateration, and scene analysis [3], we use the triangulation and trilateration approach that exploit geometric properties of a given scenario. Using elementary geometry, distances can be used to derive information about node locations. Trilateration and triangulation methods are based on distances between entities and angles between nodes, respectively. To estimate the locations on a plane, at least three non-collinear anchor points are required. Using the distances (between anchor points and the nodes) and anchor nodes' location, the node positions have to be at the intersection point of the three anchor nodes' transmission ranges. This basic fact is illustrated in Fig. 2. By moving a mobile sink adequately we can obtain the required anchor points for SOSP.

In order to apply the (multi-) lateration method, estimates for the distances from the node to the anchor points are required. This information can be obtained using Time of Arrival (TOA), Time Difference of Arrival (TDOA), Angle of Arrival (AOA) or Received Signal Strength Indicator (RSSI) [3]. In our algorithm, we use TOA as a distance estimation technique. This is also known as time of flight method. The TOA exploits the conversion of the distance from the transmission time, when the propagation speed is known. Upon receiving a small packet from a sender, a receiver immediately returns the packet to the sender. Assuming the same forward and backward paths, the sender measures the round trip time and estimates the distance from this. The distances obtained by TOA are then used in trilateration to estimate the nodes' locations. We assume that all computations are done by the mobile sink, and thus do not constitute a burden for the sensor nodes.

Fig. 2 illustrates the trilateration process for a node location with 3 anchor points. Knowing the locations of three anchor points, (x_i, y_i), we can estimate the distance d_i where $i = 1, 2, 3$. Using elementary geometry, we can calculate the location of the node position.

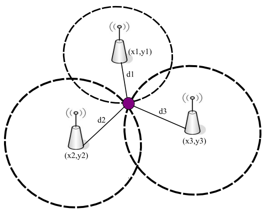

Figure 2: Trilateration with 3 anchor points.

3.3.2 *Determination of n-Hop Values*

The purpose of calculating n-hop values is to determine the maximum number of hops necessary for grouping without endangering connectivity of the network. In fact, the determination of n-hop values is not easily predictable due to several factors on which it depends, such as the number of nodes and sinks, transmission range, and node density. Also, the actual sink location affects the value of n. However, a priori knowledge of the n-hop value helps minimizing energy consumption while keeping balance between groups with respect to the number of nodes. Actually, it is not a fundamental necessity to be balanced, and, in fact, in some configurations it might be better to follow the nearest sink rule for routing in order to minimize the maximum worst-case delay. On the other hand, most often an unbalanced network produces traffic hot spots and diminishes the system performance as well as the lifetime of the WSNs. Therefore, we analyze n-hop values under varying parameters and determine typical n-hop values for small-scale and large-scale WSNs. Simulative results for n-hop values are shown in the following tables together with their related factors. We analyze n-hop values with sink to node ratios of 1:50 and 1:100. The results are given as the minimum and maximum n-hop values over 10 different random node distributions. Having the same ratio of nodes and sinks, the required n-hop distances are quite stable. Here, we use the same experimental setup as in Section 4.3.

3.4 Sink Location Selection

The step following the grouping phase is to choose the optimal sink placement. The goal of SOSP is to design an

Table 1: Table of n-hop values with sink to node ratio of 1:50.

nodes	sinks	txRange	density	(min,max) n-hop
50	1	$16m$	$0.01\frac{nodes}{m^2}$	$(4, 6)$
100	2	$16m$	$0.01\frac{nodes}{m^2}$	$(5, 6)$
200	4	$16m$	$0.01\frac{nodes}{m^2}$	$(6, 7)$
400	8	$16m$	$0.01\frac{nodes}{m^2}$	$(6, 7)$

Table 2: Table of n-hop values with sink to node ratio of 1:100.

nodes	sinks	txRange	density	(min,max) n-hop
100	1	$18m$	$0.01\frac{nodes}{m^2}$	$(6, 8)$
200	2	$18m$	$0.01\frac{nodes}{m^2}$	$(7, 8)$
400	4	$18m$	$0.01\frac{nodes}{m^2}$	$(7, 9)$

algorithm for the sink placement which minimizes the maximum worst-case delay. At the same time, the algorithm should have an acceptable computation and communication effort for an on-line operation. Due to these design criteria we focus on the area near CGSC, especially within the 1-hop neighbor region, for sink location selection. As we face a large, continuous search space, the computation time for an on-line operation of SOSP indicates a purely local search. Because of the continuous search space, we first discretize and determine a set of candidate locations for sink placement. The following subsection explains the way to compute the candidate locations.

3.4.1 *Determination of Candidate Locations*

We propose two methods of a candidate location selection. The first method is based on the discretization of the originally continuous search space into a finite search space [7], whereas the latter is based on the fixed set of candidate locations. In order to determine the candidate locations within the 1-hop neighborhood we must know the 1-hop neighbor nodes' locations which are obtained from the grouping phase for both methods. The first method simply samples the search space and chooses a candidate from each indifference region. Using the sensor nodes' locations and their transmission ranges, the sampling takes place over all possible regions and collects a point from each of them. Fig. 3(a) illlustrates all possible regions for sink placement. From the intersection of the nodes' transmission ranges we obtain a tesselation of the sensor field. The atomic regions forming that tesselation become the indifference region. Note that no matter where we place a sink within such an area the routing topology will not be altered, which is why we can just choose any point inside such a region as a candidate location. For further details see [8]. If it were for a global search, the whole space would need to be discretized in that fashion, however, for our purposes we discretize only the area of the 1-hop neighbors' transmissions intersection. Therefore, this method needs the nodes' locations and transmission ranges to determine the candidate locations. This means that candidate locations may vary with respect to the number of neighbors and their locations. Although this method guarantees confines to the 1-hop neighborhood, the computational effort for sampling is still pretty high and may strain the on-line operation of SOSP.

As opposed to the sampling method, we introduce a fixed

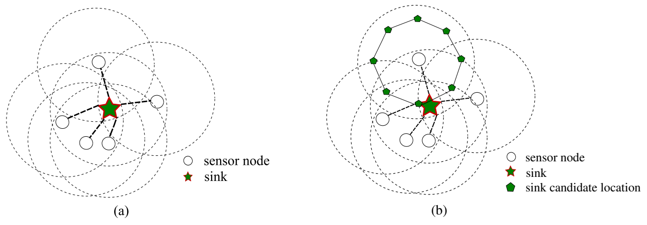

Figure 3: (a) Discretizing of candidate locations among 1-hop neighbors and (b) fixed candidate locations at circumradius of a regular octagon.

candidate location generation which has considerably lower communication overhead and computational effort. Again, we focus on the 1-hop neighbors' positions and their transmission ranges in this method. Instead of sampling over the search space, we place the candidate locations at fixed points inside the transmission range of each neighbor node. The fixed candidate locations are placed at the corners of a regular octagon as illustrated in Fig. 3(b). Those points can easily be calculated from the sensor nodes' locations. We use three-quarters of the node's transmission range as a distance between the node and the candidate location (circumradius of octagon). For example, we place a candidate location at 12m distance (which is three-quarters of our typically assumed 16m transmission range) from the node at a corner of the regular octagon. The closer this distance becomes to the transmission range the better the chance for more 1-hop neighbors. As illustrated in Fig. 3(b), some candidate locations may be duplicated with respect to covering indifference regions. Although this is unavoidable, it must not constitute a bad thing since the indifference regions may be different when factoring in 2-hop neighbors. For this reason, we also consider the distance that is as far as possible from the node location. In this method, the number of the candidate locations depends only upon the number of neighbor nodes. In particular, the total candidate locations will be eight times the number of 1-hop neighbors if the fixed candidate is based on a regular octagon shape.

The following histogram shows a quantitative comparison between fixed and sampled candidate locations in the SOSP strategy. We analyze this experiment based on the assumptions from Section 4.3.

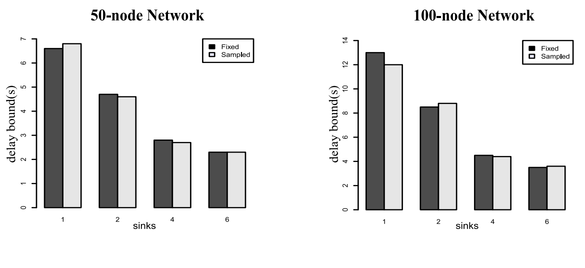

Figure 4: Comparison between fixed and sampled candidate locations in SOSP.

As shown in Fig. 4, the fixed candidate location scheme performs almost as good as the exhaustive sampling scheme. With the advantages of lower communication overhead and computational expensiveness we further on opted for the fixed candidate locations scheme in SOSP.

When knowing the candidate location set for sink placement, we can start with the sink location selection. We assume that the mobile sink traverses from one candidate location to another and computes the maximum worst-case delays which are the function of the routing topology. After these calculations, the sink location minimizing the maximum worst-case delay is selected. All of these actions can be done in parallel for each of the mobile sinks.

3.5 Operation

After selecting the best sink location in each group, the sinks are made stationary at those locations. In fact, it is well conceivable that sink locations are recomputed from time to time in order to deal with network changes, which would require the sinks to become mobile again, but we leave this for future work. The actual operation of the WSN can now start, for example, after broadcasting a query to all sensor nodes. A node receiving the message forwards it to the others until all nodes are reached. The sensor nodes send back the message to the nearest sink in order to fix the routing. Nearest here means the hop distance because the message transfer delay is computed over multi-hop communications. Note that this could mean a different routing from the grouping phase, where groups were build independently from one another.

Algorithm 1 summarizes all steps of SOSP.

Algorithm 1 SOSP Algorithm.

```
Given: a circular sensor field with known radius and transmission ranges of nodes and sinks.
Definitions: initial locations of sinks, $S_i$, where $i = 1, 2, ..., k$, nodes $N_j$, where $j = 1, 2, ..., N$.
1. Deployment Phase
  (i) Deploy a random node distribution
2. Grouping Phase
  (i) Place $k$ sinks at CGSCs
  (ii) Create 1-hop neighbors set for $S_i$ by transmitting a signal and whichever node replies to the signal is collected
  (iii) Determine 1-hop neighbors distances from $S_i$ and thus locations by using trilateration with TOA and 3 anchor points
  (vi) Group up to $n$-hop distance ($n$-value is obtained by simulation)
3. Sink Location Selection Phase
   For each group, ($k$ sinks represent $k$ groups)
  (i) Determine the fixed candidate locations according to the 1-hop neighbors' locations set from 2(iii)
  (ii) A mobile sink traverses into each candidate location and calculates the maximum worst-case delay
  (iv) Select the best sink, i.e., the one minimizing the maximum worst-case delay
4. Operation Phase
   Upon the selection of the best sink from each group,
  (i) Allow $N_j$ to connect to the nearest (i.e., the shortest hop distance) sink
  (ii) Calculate the maximum worst-case delay
```

In our algorithm, the only information we need is the field size, the transmission ranges of both sensors nodes and sinks, and the number of nodes. The step following the random node deployment phase is grouping. After assigning each node to a group, we can start with the sink placement phase. The fixed assignment of candidate location sets is calculated based on the location information obtained from the grouping phase. Note that we only use the location information of 1-hop neighbors. After the candidate location sets have been determined, a mobile sink traverses from one candidate location to another, computes the maximum worst-case delay, and finally picks the location having the minimum value. We assume that the mobile sink has more than enough energy supply for these operations. The nodes do not consume considerable amounts of energy due to the design of SOSP. Upon achieving the best self-organized sink placement for each group, we can start the operation and then calculate the actual global maximum worst-case delay. Since we have a fully connected network, there is at least one sink for each of the nodes. In order to obtain a better performance, the nodes are allowed to connect to the nearest sink (i.e., the minimum hop distance sink) in SOSP. This approach is helpful for the worst-case delay performance, since the message transfer delay is strongly affected by greater hop distances and aggregate flows toward the sink.

4. PERFORMANCE EVALUATION

In this section we describe the performance evaluation of SOSP. Before its presentation, we introduce some facts that are used in the evaluation. We refer to our previous works [7, 8] several times since we use those techniques as benchmarks for the new SOSP algorithm. Therefore, we discuss some basic concepts from our previous works for the purpose of better understanding. Besides, the worst-case network analysis framework, sensor network calculus, being used is also presented in brief.

4.1 Introduction of GSP and GASP Strategies

The Geographic Sink Placement (GSP) is intended as a very simple, almost fully scenario-agnostic sink placement strategy. The only information required is the size of the sensor field (as with SOSP). GSP is designed to work well for uniformly distributed sensor nodes and does not require any information about the actual sensor nodes' locations. For any number of sensor nodes, it requires only the number of sinks to be deployed and the radius of the network field in order to calculate the CGSC. The center of gravity of a sector with angle α always lies on the middle radial line ($\alpha/2$) of that sector. The formulation of CGSC is given in Equations 1 and 2.

$$CGSC = F(\alpha) \times R \tag{1}$$

$$F(\alpha) = \frac{\frac{4}{3}sin(\frac{\alpha}{2})}{\alpha} \tag{2}$$

where α is in radians, $0 \leq \alpha \leq 2\pi$, and R is the radius.

As discussed in [7], the GSP strategy gives pretty good results for uniform node distributions. It is as lightweight as one could imagine and with respect to this has an edge over all other strategies.

At the other end of the spectrum compared to GSP, GASP represents a Genetic Algorithm (GA)-based heuristic method for sink placement. Although it cannot absolutely guarantee to find an optimal solution, GASP has shown itself to be a very good heuristic on achieving near-optimal sink placements. The candidate locations are obtained by discretizing the search space without loosing optimality. Each candidate location represents the so-called indifference region, in which, no matter where we place a sink, the routing topology together with the value of the objective function, the maximum worst-case delay, stay unchanged. After establishing the candidate locations set, GASP is applied. To implement GASP, the first step is to select initial individuals from the overall search space (i.e., the whole set of candidate locations). Each individual represents a solution to a problem and is coded in the so-called "chromosome", where we chose a problem-specific good representation. The selected set of individuals becomes the initial "population". Next, the GA operators, crossover and mutation are all implemented, and problem-specifically, together with the "fitness" are also computed for each individual. After that the final operator of the GA, the selection, is invoked to create the next "generation" of individuals that forms a new population. The termination of the GASP is done after a number of evolutionary loops. Since the cardinality of the candidate locations grows rapidly for larger networks, the number of all possible combinations, and therefore the search space, is vast, making total enumeration unthinkable. The GA-based enumeration process requires considerably lower computational effort while providing, at least for smaller instances, almost always the globally optimal sink placement [8] (for larger instances the global optima cannot be computed). A clear drawback of the GASP is that it requires global knowledge about a sensor network, in particular, all the node locations, and that it makes further assumptions that are inconvenient for large-scale WSNs. Furthermore, it is still quite compute-intensive. These are actually the reasons why we in this paper proposed SOSP for large-scale WSNs.

4.2 Minimizing the Maximum Worst-Case Delay

As we mentioned before, network calculus [5] guarantees as a framework for the worst-case analysis in WSNs in [10].

In [10], bounding processes called arrival curve α and service curves β can capture the major worst-case properties for data flows: maximum delay and maximum backlog. Here are some definitions: the arrival curve bounds the input function, which is the sensed data of every sensor node, whereas the service curve depends on the duty cycle, and can therefore be adjusted to achieve certain energy-efficiency goals. The heart of sensor network calculus is formed by three bounds: backlog bound, delay bound, and output bound. In general, the backlog bound is the vertical deviation between α and β and the delay bound is the horizontal deviation between α and β. The output bound of each server can be calculated by using deconvoluting arrival and service curves $\alpha \oslash \beta$. Detailed explanations of the SNC can be found in [10, 14], and [15].

Based on these definitions, delay, backlog and output bounds can be calculated with different analysis methods. We use the so-called Pay Multiplexing Only Once analysis (PMOO) [13], which is shown to deliver a tight bound for sink-trees [12] of homogeneous nodes, to calculate the end-to-end delay for each flow of interest from one sensor node to the sink.

4.3 Experimental Results

All experiments are based on the following assumptions:

A circular field shaped network is chosen for initial sink placement at CGSC. The sensor nodes are deployed with a density of $0.01\frac{nodes}{m^2}$ in uniform random node distribution fashion. The transmission range is $16m$ for both sensor and sink nodes. We analyze scenarios of 100, 200 and 500 node networks with 2-6 sinks. In each scenario, we analyze 10 different node distributions and take the average of their results to counter random effects; in fact, in none of the comparisons below, 99% confidence intervals were even near to overlapping. The same network scenarios are used for each strategy. The routing we use here is based on the Dijkstra's shortest path. Under the homogeneous nodes assumption, the token bucket arrival curves and rate-latency service curves are considered for the network calculus operations. In particular, for the service curve we use rate-latency functions that correspond to a duty cycle of 1% and 11.5%[1] depending on the network size, since for larger networks a duty cycle of 1% results in infinite delay bounds. For example, a duty cycle of 1% results in a latency of 1.096s and a forwarding rate of 258 b/s. As mentioned above, we use PMOO network analysis.

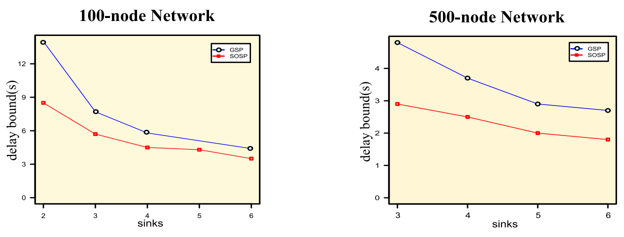

Figure 5: The worst-case delay comparison of SOSP vs. GSP.

4.3.1 *Performance Comparison of SOSP vs. GSP*

At first, we evaluate the performance between SOSP and GSP strategies. The GSP sink placement is used as an initial sink placement in the SOSP algorithm, so the later always has to outperform GSP. Furthermore, SOSP uses mobile sinks for a self-organized network operation and should therefore, have an edge over GSP in terms of delay minimization.

In fact, as expected, SOSP outperforms the GSP strategy as shown in Fig. 5. In a 100 node network, the worst-case delay of GSP improves from 14 to 7.7 to 5.8 to 5.1 to 4.4s for the 2- to 6-sink scenarios, respectively. For SOSP, the worst-case delay improves from 8.5 to 5.7 to 4.5 to 4.3 to 3.5s for the 2- to 6-sink scenarios, respectively. The delay differences between GSP and SOSP vary from 0.8s to 5.5s. So, for example, to provide the same delay performance to SOSP with 4 sinks, 6 sinks for the GSP are required. In our experiments, we noticed that the more sinks, the smaller the worst-case delay gap between GSP and SOSP strategies.

For 500 nodes, the worst-case delay gaps between the two strategies are 1.9s, 1.2s, 0.9s and 0.9s for 3 to 6 sinks placement, respectively. The reason for having a small delay gap is the node distribution pattern and a higher duty cycle. The higher duty cycle results in a lower latency and a higher forwarding rate, thus providing a faster message transfer delay. Accordingly, the worst-case delay gap becomes smaller. But note again that obtaining the same delay performance as SOSP with 3 sinks would require 6 sinks for the GSP.

In fact, all investigated scenarios are based on uniform random node distributions, which is a good "playground" for the GSP strategy. We, therefore, performed another experiment under non-uniform random node distribution in order to check whether SOSP can outperform GSP strategy more pronouncedly.

In the non-uniform random node distribution network, the node density varies from region to region. Nodes are densely deployed in some regions but not in all. The sink placement in GSP is fixed at CGSC and does not depend on the node distribution pattern. Thus, GSP should be expected to produce higher worst-case delays in non-uniform random node distributions. In opposite to GSP, SOSP should perform well in non-uniform random node distributions. Whatever node distribution pattern is applied, SOSP self-organizes a good sink placement which has a low maximum worst-case delay. In non-uniform node distributions, some candidate locations from regions with lower density may provide good candidates for regions with higher density. This means that candidate locations near regions of higher density can offload the respective sink in those regions to achieve better delay performance. With this flexible movement, SOSP is very robust against different node distribution patterns.

Fig. 6 shows how SOSP outperforms GSP under non-uniform random node distribution. For a fully connected

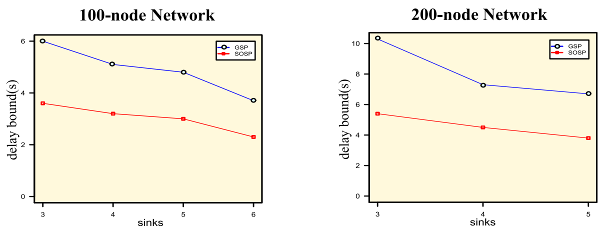

Figure 6: SOSP vs. GSP in non-uniform network.

network, the transmission ranges for both sensor nodes and sink have to be increased in non-uniform random node distributions. A 22m transmission range is used for 100 and 200 node networks with 1% duty cycle. We analyze 3 to 6 sinks scenarios for both networks since 1% duty cycle results in infinite delay bounds for the 200 node network with 2 sinks.

In the 100 node network, the delay gaps improve from 2.4s, 1.9s, 1.8s, and 1.4s for 3 to 6 sinks scenarios, respectively. For 200 nodes, the delay gaps improve from 4.9s, 2.8s, 2.9s, and 2.1s from 3 to 6 sinks scenarios. These results show that SOSP considerably outperforms GSP in non-uniform networks. For example, for both networks SOSP with 3 sinks outperforms GSP even if that one is given 6 sinks.

4.3.2 *Performance Comparison of SOSP vs. GASP*

As we introduced the heuristic GASP strategy [8] for near-optimal sink placement, we can use it for benchmarking SOSP against a good global strategy. The performance evaluation of SOSP and GASP, along with the results for GSP, is shown in Fig. 7. In particular, we analyze the scenario for a 100 node network with 4 sinks. We restricted the scenario to 100 nodes due to the considerable amount of computations for larger networks with the GASP. The result is again the average over 10 different node distributions. For the GASP strategy, each evaluation consists of a population size of 80 individuals and the number of generations was set to 100,

[1]The values are based on a realistic node model of a Mica2 mote running the TinyOS system (see CC1000 Radio Stack Mannual) [1].

resulting in 8000 different sink placements. In comparison, SOSP uses about 300 different sink placements for choosing the best location.

The worst-case delay improves from 5.8s to 4.5s to 4.2s for GSP, SOSP, and GASP, respectively. The performance of the SOSP strategy is close to that of the GASP strategy, which can be considered a success.

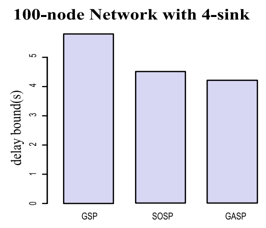

Figure 7: The worst-case delay comparison among GSP, SOSP and GASP.

5. CONCLUSION

In this paper, we introduced a self-organized sink placement algorithm with lower computation and communication overhead compared to previous solutions to minimize the maximum worst-case delay in WSNs. This algorithm was inspired by our previous works [7] and [8], of which we used the respective advantages. In particular, we put emphasis on the self-organized nature of our sink placement algorithm without using global knowledge as, e.g., the sensor nodes' locations. We consider this key for an application in large-scale WSNs. In particular, as we require only information from the 1-hop neighborhood of the initial sink placement (at CGSC), the algorithm, SOSP, should scale up to very large WSNs. From the experimental results, it is clear that SOSP clearly outperforms an almost totally scenario-agnostic strategy (GSP) and comes very close to a global, near-optimal strategy (GASP), which however does not scale. In conclusion, SOSP strategy is shown to be a promising scalable solution to the sink placement problem with low computation and communication overhead.

6. ACKNOWLEDGMENT

This work has been generously supported by DFG under grant SCHM2528/2-1 and Gottlieb Daimler- und Karl Benz-Stiftung under grant number 02-13/06, 2009. The authors are grateful to Petra Martinovic for proofreading.

7. REFERENCES

[1] http://www.tinyos.net/tinyos-1.x/doc/mica2radio/cc1000.html.

[2] Chen, C., Ma, J., and Yu, K. Designing Energy-Efficient Wireless Sensor Networks with Mobile Sinks. In *Proceeding of the 4th ACM Conference on Embedded Networked Sensor Systems (SenSys 2006)* (Boulder, Colorado, USA., Oct. 2006).

[3] Karl, H., and Wittig., A. *Protocols and Architectures for Wireless Sensor Networks.* Wiley, 2005.

[4] Kim, H., Seok, Y., Choi, N., Choi, Y., and Kwon, T. Optimal Multi-sink Positioning and Energy-efficient Routing in Wireless Sensor Networks. In *Lecture Notes in Computer Science(LNCS)* (2005), vol. 3391, pp. 264–274.

[5] Le Boudec, J.-Y., and Thiran, P. *Network Calculus - A Theory of Deterministic Queuing Systems for the Internet.* Springer, 2001.

[6] Oyman, E. I., and Ersoy, C. Multiple sink network design problem in large scale wireless networks. In *IEEE International Conference on Communications (ICC)* (June 2004).

[7] Poe, W. Y., and Schmitt, J. B. Minimizing the Maximum Delay in Wireless Sensor Networks by Intelligent Sink Placements. Technical Report 362/07, University of Kaiserslautern, Germany, July 2007.

[8] Poe, W. Y., and Schmitt, J. B. Placing Multiple Sinks in Time-Sensitive Wireless Sensor Networks using a Genetic Algorithm. In *14th GI/ITG Conference on Measurement, Modeling, and Evaluation of Computer and Communication Systems (MMB 2008)* (Mar. 2008), GI/ITG.

[9] Poe, W. Y., and Schmitt, J. B. Self-Organized Sink Placement in Large-Scale Wireless Sensor Networks. In *17th Annual Meeting of the IEEE International Symposium on Modelling, Analysis and Simulation of Computer and Telecommunication Systems (MASCOTS 2009)* (Sept. 2009), IEEE.

[10] Schmitt, J. B., and Roedig, U. Sensor Network Calculus - A Framework for Worst Case Analysis. In *Proceedings of the International Conference on Distributed Computing in Sensor System (DCOSS05)* (June 2005).

[11] Schmitt, J. B., and Zdarsky, F. A. The DISCO Network Calculator - A Toolbox for Worst Case Analysis. In *Proceeding of the First International Conference on Performance Evaluation Methodologies and Tools (VALUETOOLS'06)* (Nov. 2006), ACM.

[12] Schmitt, J. B., Zdarsky, F. A., and Fidler, M. Delay bounds under arbitrary aggregate multiplexing: When network calculus leaves you in the lurch... In *Proceedings of the IEEE INFOCOM* (Apr. 2008).

[13] Schmitt, J. B., Zdarsky, F. A., and Martinovic, I. Performance Bounds in Feed-Forward Networks under Blind Multiplexing. Technical Report 349/06, University of Kaiserslautern, Germany, Apr. 2006.

[14] Schmitt, J. B., Zdarsky, F. A., and Roedig, U. Sensor Network Calculus with Multiple Sinks. In *Proceedings of the Performance Control in Wireless Sensor Networks Workshop at the 2006 IFIP Networking Conference* (May 2006).

[15] Schmitt, J. B., Zdarsky, F. A., and Thiele, L. A Comprehensive Worst-Case Calculus for Wireless Sensor Networks with In-Network Processing. In *IEEE Real-Time Systems Symposium (RTSS'07)* (Tucson, AZ, USA, Dec. 2007).

[16] Vincze, Z., Fodor, K., Vida, R., and Vidacs, A. Electrostatic Modelling of Multiple Mobile Sinks in Wireless Sensor Networks. In *Proceedings of the Performance Control in Wireless Sensor Networks Workshop at the 2006 IFIP Networking Conference* (May 2006).

[17] Xu, R., and II, D. W. Survey of Clustering Algorithms. In *IEEE Transactions on Neural Networks* (May 2005).