Fair Capacity Sharing Among Multiple Sinks in Wireless Sensor Networks

Bing Han
GET - ENST Bretagne - ENST
Brest - Paris, France
Beijing University of Posts and Telecommunications
Beijing, China
bing.han@enst-bretagne.fr

Gwendal Simon
GET - ENST Bretagne
Brest, France
gwendal.simon@enst-bretagne.fr

Abstract

The data transmission capacity of wireless sensor networks is known to be limited by both the hardware and the sink-based communication paradigm. In a multi-sink application scenario, simultaneous queries may generate traffic exceeding the transmission capacity of certain sensor nodes. In this paper, we intend to share the capacity of the sensors among multiple sinks by adjusting their query ranges, so that no sensor gets congested and every sink is able to monitor an area with desired data rate. Specifically, the max-min fairness will be ensured. An analytical model is presented and theoretical results for two-sink case is given. Then a distributed algorithm allowing sinks to approach the optimal query range with local knowledge is developed. Finally, simulation results are shown to illustrate the characteristics of this capacity sharing problem and to validate the effectiveness of our distributed algorithm.

1. Introduction

In certain critical areas such as those regulated by the Seveso directive, the density of sensors is expected to be high enough to build a self-organized multi-hop wireless sensor network (WSN) [1]. During crisis, such networks could substantially help the intervention teams by notifying selected events. A typical application of this network could be a monitoring system with some device-equipped firemen gathering data from a burning area in order to determine a safe perimeter, while others operating on the hearth are under real-time alert about risks of nearby explosions. The firemen send requests to and collect data from the sensors within a specific area in a multi-hop fashion. Thus they could be seen as *mobile sinks*.

This monitoring system should be trustworthy so that all events within a monitored area must be reported to the querying sink, and should be adaptable since the number of sinks changes over time. Besides, it is desired for each sink to monitor the largest possible area to ensure individual security. However, since the sensor nodes have limited data transmission capability, multiple data requests could congest sensors in certain area, depending on the position of the sinks, the size of the requested area, etc. Furthermore, if the sensors are already saturated by current queries, newly joined sinks will be kept in a starved state and unable to retrieve information from those sensors until some of the existing sinks leave. On the contrary, if each fireman accepts to decrease the size of its query area, it may enable newcomers to use the network as well. Thus, fairness among sinks when sharing the network is preferred. As a result, the sinks tend to increase their queries as long as some fairness rules are kept. We assume that sinks do not coordinate through direct communication links because this will form a complex network structure which will be studied in future work.

The contributions of this paper are threefold. First, we introduce a simple model for the capacity sharing problem. We give a formal definition of *supported sinks* and *feasible network configurations*, then we describe immediate results obtained from previous works. Second, we present a distributed algorithm allowing sinks to determine their query radii with respect to nearby sensors and sinks. The algorithm is designed to be adaptive to the number of sinks. Finally, some related issues are illustrated and the validity of the algorithms proposed is justified with simulations.

This paper is organized as follows. The problem is

This work is funded by Le Groupe des Écoles des Télécommunications (GET), France, under the "Programme Initiative: Réseaux Autonomes et Spontanés".

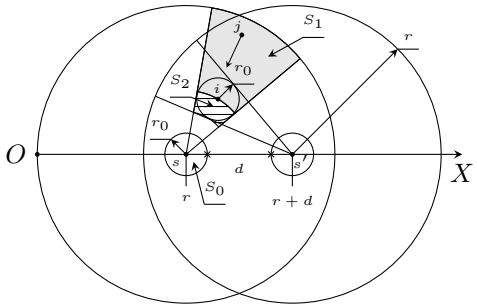


Figure 1. Traffic load model.

formally defined and the notations are introduced in Section 2. Theoretical results when only two sinks share the network are derived in Section 3. Then Section 4 describes a distributed algorithm which will be analyzed extensively by simulations in Section 5. We review the related works in Section 6 before drawing a conclusion in Section 7.

2. System model and notations

In this section, we introduce our system model with the definitions and notations. Some of them with geometrical meanings are presented in Figure 1.

Wireless sensor network: A WSN consists of a set $\mathcal{V}$ of n sensor nodes deployed in a fixed area according to a Poisson distribution with density λ_0 and a set $\mathcal{S}$ of m sink nodes. Each node is assumed to have an omnidirectional wireless transceiver that is able to transmit or receive W bits of data per second to or from a fixed maximum distance $r_0 > 0$. Nodes are unable to receive simultaneous messages. It is natural to assume the wireless transmission range is unable to cover the whole network, thus for the communications between sensors and sinks, a multi-hop routing protocol has to be employed. Furthermore, we assume the routing protocol ensures shortest path and load balancing for packets. Finally, we do not consider malicious behaviors in this study.

Query model: A *query* emitted by a sink $s \in S$ pertains to a sub-area assumed to be a disk of radius r centered at the sink and will be referred to as $Q_{rb}(s)$, where b denotes that each sensor within the queried area will generate b bits of data per second in response to this query. We assume there is no aggregation or compression when the data is collected and restrict each sink always has one running query, thus there are exactly m running queries in the network.

We say that a sensor is covered by a query if it generates data for the query. The set of sensors covered by query $Q_{rb}(s)$ is noted as $\mathcal{V}_{rb}(s) \subseteq \mathcal{V}$. A sensor may be covered by more than one queries and the set of queries sensor p has to process is noted as $\mathcal{Q}(p)$. The circle with radius r_0 around the sink s is noted as $S_0(s)$, as shown in Figure 1, and will also be referred to as the *critical region* of s, since it is usually the busiest region within the query. Sink s is a (r, b)-*supported sink* by the network if for a query $Q_{rb}(s)$, all data emitted by sensors in $\mathcal{V}_{rb}(s)$ arrive to s.

Obviously, the positions of sensors and sinks are necessary for this kind of area based query model. However, GPS capability is not mandatory since several localization protocols are available, *e.g.* [2], as a result of intensive studies on such mechanisms.

Traffic load model: The traffic load model we will employ has been proposed in [3] and extended in [4]. The authors of [3] observed that for two sensors i, j and a sink s such that j is outside the one-hop region $S_0(s)$ of the sink, the data generated by j destined to s is forwarded by i if the distance from i to the line $\overline{js}$ is less than r_0 and the projection of i on $\overline{js}$ lies between j and s. Figure 1 is largely depicted from [4], where s and s' are sinks and i and j are two representative sensor nodes. Two cross marks denote the most loaded points. When sensor i is outside $S_0(s)$, it is responsible for forwarding the data generated by all sensors within the sector S_1 behind it (gray shadowed sector of ring in Figure 1), whose sides are the tangents from the sink to the circle centered at sensor i with radius r_0. Since we assumed that the underlying routing protocol would provide us with shortest path and load balancing properties, a single sensor i is unlikely to handle all the traffic from S_1. Rather, the traffic load from both S_1 and S_2 regions will be shared by all sensors within S_2 region. Let d_{si} denote the distance between nodes s and i, the region S_2 is a sector of ring whose inner and outer radii are $d_{si} - r_0$ and d_{si}, respectively, and whose sides are the same tangents as that of S_1, as shown by the area shadowed with horizontal line pattern in Figure 1. Thus, the average traffic load of i is proportional to $(S_1 + S_2)/S_2$. On the other hand, sensors within the $S_0(s)$ will share the traffic from all over the network. Let $\delta(d_{ps})$ the traffic load of a sensor p at distance d_{ps} to a sink s, we have:

$$\delta(d_{ps}) = \begin{cases} \frac{br^2}{r_0^2} & p \in S_0(s) \\ b + \frac{2br^2}{\pi r_0^2} \arcsin\left(\frac{r_0}{d_{ps}}\right) & \\ \quad - \frac{2bd_{ps}^2}{\pi r_0^2} \arcsin\left(\frac{r_0}{d_{ps}}\right) & p \in Q_{rb}(s) \backslash S_0(s) \\ 0 & \text{otherwise} \end{cases} \quad (1)$$

Max-min fair configuration: A *configuration* is defined by a set of radius-sink pairs: $C = \{(r_0, s_0), (r_1, s_1), \ldots, (r_m, s_m)\}$ and we say C is a

feasible configuration if $\forall(r_i, s_i) \in C$, s_i is (r_i, b)-supported. Although there exists many feasible configurations for a given network topology, we are interested in the one with *max-min fair* properties such that smallest radii are maximized, then the second smaller radii are maximized and so on. A basic rule is that maximizing a query should not shrink other already smaller queries. Obviously, the capacity constraint should be kept at the same time. Note that in the context we are considering, the traffic b is fixed and each sink is interested in maximizing its query radius to maximize individual security.

3. Analysis for two-sink case

Now we consider only two sinks s and s' are in the network. This simple case provides a good introduction to our capacity sharing problem. Intuitively, the max-min fair radius for each sink should take the same value under this case. We distinguish three cases based on the relative position of the two sinks: distant sinks, not-so-distant sinks and nearby sinks. For each case, we shall analyze the traffic load and then derive the max-min fair query radius for each sink.

Distant sinks: In a WSN with Poisson node distribution and all-to-one communication paradigm, the bottleneck is the sink itself [5]. We say that a query $Q_{rb}(s)$ is *globally maximized* when the bandwidth of all nodes within $S_0(s)$ are saturated by traffic exclusively dedicated to s. If we ignore the bandwidth consumed by protocol overheads, query $Q_{rb}(s)$ is globally maximized when $\pi r^2 \lambda_0 b = W$. When the queried data rate b and the density of sensors λ_0 are fixed, we naturally obtain:

$$r_{max} = \sqrt{\frac{W}{\pi b \lambda_0}}$$

The sinks s and s' are considered as *distant* when the S_0 area of one query does not overlap with the other query. This happens when $d_{ss'} > r_{max} + r_0$. In this case, the maximum amount of data a sink can receive is bounded by its bandwidth and the two queries could be both globally maximized with the same radius r_{max}. Obviously, the configuration $C = \{(r_{max}, s), (r_{max}, s')\}$ is max-min fair.

Not-so-distant sinks: When $2r_0 < d_{ss'} \leq r_{max} + r_0$, the S_0 area of one query may be covered by the other query but $S_0(s)$ and $S_0(s')$ do not overlap. Under this case, if either of the two queries is globally maximized, the other query has to shrink accordingly. On the other hand, the two sinks may coordinate to achieve an equilibrium state such that they have the same query radius. This state is exactly the max-min fairness we want to achieve. Under this configuration, both sinks s and s' experience similar queries and sensor throughput. The total traffic handled by a sensor p is $\delta(d_{ps}) + \delta(d_{ps'})$. Due to the bottleneck effect of the sink, the total traffic of the sensors within the S_0 region of either sink should not exceed the bandwidth of the sink. Note that the traffic generated or forwarded by sensors in $S_0(s)$, whether it is for s or s', consumes the bandwidth of s because all sensors in $S_0(s)$ share the wireless medium with s. Thus, in order to avoid congestion within S_0 region, the sum traffic in this region should always be kept under W. The calculation of the sum traffic within S_0 region requires integration of the traffic load function (1) on S_0, which gives no explicit solution of the max-min fair query radius. However, the characteristics of the traffic function produces two most loaded points lying at the intersection of the line joining s and s' and the two circles delimiting $S_0(s)$ and $S_0(s')$, as shown in Figure 1 with two cross marks. If we setup a coordinate system with X axis and origin O as shown in Figure 1, the traffic load of sensors on X could be ploted in Figure 2, where two sinks lie at $x = r$ and $x = r + d$ and we see two maximum at $x = r + r_0$ and $x = r + d - r_0$. Note that the traffic load for other sensors not on X is always lighter than those on X. As a result, we could limit the traffic of these two busiest sensors under the *shared* bandwidth of sensors within S_0 to make sure that no congestion occurs.

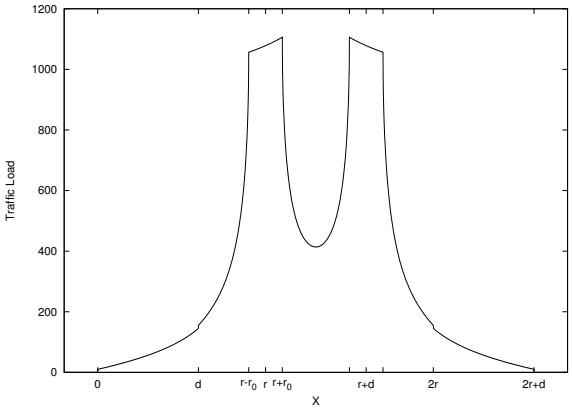


Figure 2. Traffic load for sensors along X axis.

The shared bandwidth of sensors within $S_0(s)$ is $\frac{W}{\pi r_0^2 \lambda_0}$. Together with (1), the total traffic $\Delta(p)$ handled by a sensor p, when maximized with max-min fairness, should be:

$$\Delta(p) = \delta(r_0) + \delta(d_{ss'} - r_0) = \frac{W}{\pi r_0^2 \lambda_0}$$

Solving this equation for r, we obtain a common

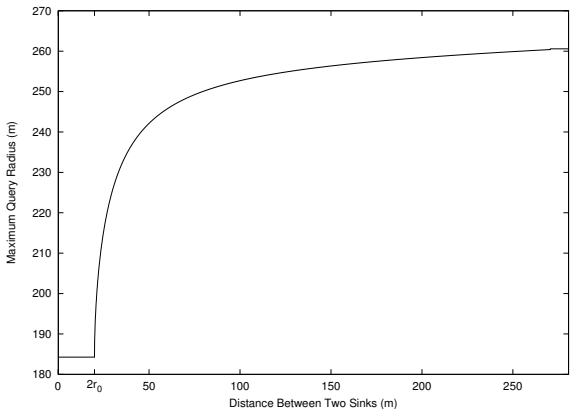


Figure 3. Common maximum query radius.

radius for both sinks, noted as r_c:

$$r_c = \sqrt{\frac{\left(\frac{W}{\pi r_0^2 \lambda_0} - b + C_1\right)}{C_2}}$$

where:

$$\begin{aligned} C_1 &= \frac{2b(d-r_0)^2}{\pi r_0^2} \arcsin\left(\frac{r_0}{d-r_0}\right) \\ C_2 &= \frac{b}{r_0^2} + \frac{2b}{\pi r_0^2} \arcsin\left(\frac{r_0}{d-r_0}\right) \end{aligned}$$

As a result, two sinks can be $(r_c, b)-$supported and each sink is able to determine r_c by communicating with a sensor within its query. Thus the max-min configuration for the not-so-distant case is $C = \{(r_c, s), (r_c, s')\}$.

Nearby sinks: In this final case, two sinks are even nearer, *i.e.* $d_{ss'} \leq 2r_0$, meaning that the critical regions of s and s' overlap. The reasoning is the same as in previous cases, except that all sensors in the area $S_0(s) \cap S_0(s')$ are bottlenecks. Thus, each sensor within this area should handle a total traffic load as:

$$\Delta(p) = \frac{2br^2}{r_0^2} = \frac{W}{\pi r_0^2 \lambda_0}$$

We quickly obtain:

$$r_c = \sqrt{\frac{W}{2\pi\lambda_0 b}}$$

The max-min fair configuration is $C = \{(r_c, s), (r_c, s')\}$.

We should note here that in our firemen application scenario, due to the various situations of the fire site, all of the three cases discussed above could exist. For example, firemen could operate separately in the forest on fire, or form small groups searching for survivors inside a building.

As a conclusion of this section, we present in Figure 3 the max-min radius for both sinks with respect to the distance between them. The transmission range of nodes is set to $r_0 = 10$m. We see that the query radius does not increase linearly with the distance between two sinks, instead, when the distance is short, increasing it by even a small value will substantially enlarge the common radius.

4. Algorithms for multi-sink case

In this section, we develop algorithms and a protocol to solve the max-min fairness problem under generic cases when more than two sinks are in the network.

4.1. Brutal force algorithm

It is known that there exists a naive algorithm based on brutal force to achieve max-min fairness [6]. This algorithm, we note it as OPT, works as follows: all sinks are simultaneously switched on, then increase their query radii with the same step length in a perfect synchronous manner until a sensor is saturated (this sensor will be referred to as a bottleneck sensor). When a sensor is saturated, the sinks querying this sensor stop increasing their queries, while others keep increasing. The algorithm ends when no query can increase. The optimality of this algorithm is in the sense that max-min fairness is kept and every query is either globally or locally maximized. Although this algorithm is not realistic, it is helpful to understand the behavior and evaluate the performance of other algorithms.

4.2. Inspiration from two-sink case

From the discussions on two-sink case, we know that if both sinks want to maximize their queries while keeping max-min fairness in mind, the resulting radii must converge to a common value. Based on this observation, we can imagine that when more than two sinks are in the network, each pair of sinks can agree on a max-min fair radius between them. As a result, if there are m sinks in the network, each sink will have a maximum of $m-1$ common radii with other sinks. If each sink set its radius to the minimum one of its common radii with others, no sensor will be overloaded. Moreover, this mechanism also ensures that the minimum query is maximized. However, we found this is not true for non-minimum radii.

This problem could be better explained with help of Figure 4, which illustrates a typical WSN with eight sinks with numbers as their labels. The small circles around the numbers denote S_0 regions of each sink while the larger ones denote the query boundaries and the crosses denote bottleneck sensors. Other sensors

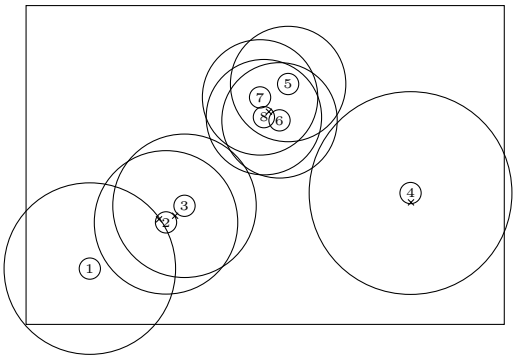


Figure 4. A WSN shared by eight sinks.

are omitted in this figure. For sink 2 and 3, the radii of their queries are bounded by a bottleneck sensor between them, so the sinks adopt the same query radius with respect to the max-min fairness policy. Now let us consider sink 1. The proximity of sink 1 with 2 let it assume a bottleneck between them which lies at one of the intersections of the boarder of $Q(1)$ and $S_0(2)$ in Figure 4. Thus, sink 1 determines a radius based on this bottleneck and *thinks* that 2 acts accordingly. But, sink 2 *does not* because the bottleneck with sink 3 is more constraining. Therefore, sink 1 is unable to achieve its maximum query radius.

Let us associate a bottleneck node $\beta(s) \in \mathcal{V} \cup \mathcal{S}$ with each query $Q_{rb}(s)$. When a sink is $(r_{\max}, b)-$supported, the bottleneck $\beta(s)$ is s itself, otherwise, the bottleneck is a certain sensor. As shown in Section 3, when there are only two sinks s and s', the two bottlenecks experience the same traffic with the same bandwidth sharing. By notation abuse, we claim $\beta(s) = \beta(s')$. For example, in Figure 4, the sinks 5, 6, 7 and 8 have the same bottleneck sensor. The problem occurs when a sink s adjusts its radius without being informed that the radius of its buddy s' does not depend on $\beta(s)$.

However, it is intuitively expected that the problem discussed above is infrequent so an algorithm based on their own bottlenecks may be sufficient for each sink. This idea inspires an algorithm, which will be referred to as LOCAL and can be stated as follows. A saturated sensor p considers itself as a potential bottleneck for all sinks in $\mathcal{Q}(p)$. Therefore, it notifies these sinks with the following information: all the radii of the queries in $\mathcal{Q}(p)$, the positions of each sink in $\mathcal{Q}(p)$ as well as the position of itself. When a sink receives such a notification, it computes a common radius with respect to all the sinks contained in the notification and adjusts its query radius to the minimum one.

An experiment was carried out over a WSN with

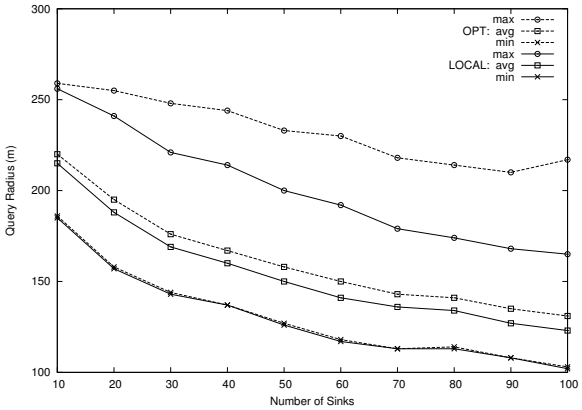


Figure 5. Query radii of OPT and LOCAL algorithms.

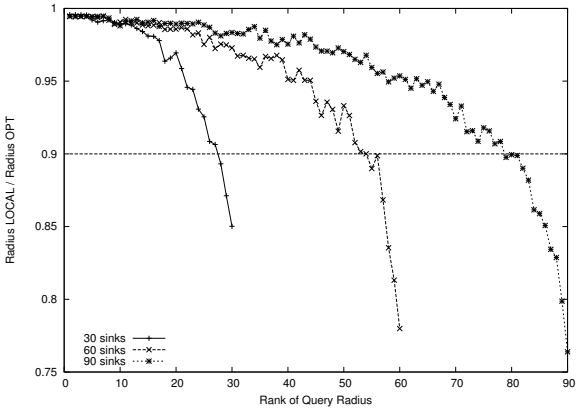


Figure 6. Ratio of LOCAL radii to OPT radii.

different number of sinks for both LOCAL and OPT algorithms. The minimal, average and maximal query radii over all sinks are plotted in Figure 5. As expected, the minimum radius are always the same for both algorithms, but average and maximum values of the LOCAL algorithm are below those of the OPT algorithm.

Another observation is illustrated in Figure 6. For three distinct contexts: 30, 60 and 90 sinks, we sort the sinks by their OPT radii in ascendant order, so that the sinks with smaller radius have a lower rank. Then for each sink, we compute the ratio between the LOCAL radii and OPT radii. On one hand, LOCAL algorithm achieves max-min fair query for sinks with minimum radius but the performance gets worse for sinks with larger radius. Under the worst cases, the ratio could be less than 0.8, indicating that the area covered by this sink would have been 1.56 times larger if OPT algorithm be used instead. On the other hand, still shown in Figure 6, only 10% of queries are notably smaller ($\leq 90\%$) than their OPT counterparts, implying that searching for OPT values based on LOCAL

algorithm could be efficient.

4.3. Distributed algorithm and protocol

Now we propose a distributed algorithm (DIS) and a protocol realizing it to achieve max-min fairness for all queries in the network by combining the two algorithms discussed in the previous section.

The idea of the algorithm is quite simple and widely known in congestion control algorithms [7]. The pseudo code is shown in Algorithm 1, where several messages carrying control information are defined: <query, s, r> message, sent to the sensors within circle (s, r) by sink s to start a query with radius r; <modify-query, s, r_{mod}> message, sent by s to the sensors within a circle whose radius equals to $\max\{r, r_{mod}\}$ to modify the query radius to r_{mod}; <saturated> message, sent by a saturated sensor p to the sinks in $\mathcal{Q}(p)$ when:

$$\Delta(p) = \sum_{s \in \mathcal{Q}(p)} \delta(d_{ps}) > \frac{W}{\pi r_0^2 \lambda_0}$$

and it carries all the radii of the queries in $\mathcal{Q}(p)$, the positions of each sink in $\mathcal{Q}(p)$ as well as the position of itself.

The algorithm consists of two phases. The first one, usually called *slow start*, is used until an approximation of the achievable radius is obtained. The radius starts at a unit value and increases according to a certain strategy until the sink is alerted by a <saturated> message. The function `initRadius` manages the initial radius increasing. On receiving the first <saturated> message, the radius is set to a lower value according to some criteria, then the system enters the second phase. In the second phase, each sink tries to increase its query radius periodically, in order to explore the OPT radius. This is handled by the `increaseRadius` function. Eventually, the sink is alerted by a <saturated> message, then it decreases its radius again and resumes increasing. The new radius is obtained by `resetRadius`.

Various implementations of `initRadius`, `increaseRadius` and `resetRadius` functions could be used. We propose one as follows. The `initRadius` is based on exponential growth, *i.e.* the query radius is initialized to 1 and increased by doubling its previous value each time it is called. The idea is to detect as quickly as possible the bottleneck sensors. Then, the `resetRadius` employs the LOCAL algorithm which returns a radius close to the optimal one in most of the cases as previously discussed. Finally, the `increaseRadius` function makes a linear increasing of the radius every k simulation rounds by a step length l. It is well-adapted because the current radius is never far from the optimal one.

Algorithm 1: Distributed Algorithm (Sink Part)

```
Data: radius
send <query, s, 1 >
while no <saturated> message do
    radius =initRadius ()
    send <query, s, radius >
end
radius =resetRadius ()
do
    while no <saturated> message do
        radius =increaseRadius ()
        send <modify-query, s, radius >
    end
    radius =resetRadius ()
    send <modify-query, s, radius >
loop
```

By choosing these implementations, the sinks see their query radii oscillate between the values of the LOCAL algorithm and the OPT algorithm. In fact, from the observation of Figure 6, it is expected that a large part of sinks experience small variations because these two values are very close. Meanwhile a small fraction of the sinks, those with the largest radius, benefit from a larger area for most of the time.

5. Simulation results

In this section, we shall present several simulation results to further illustrate the problems encountered when multiple sinks try to share the network capacity, and analyze the ability of our algorithm in solving these problems.

First of all, we describe briefly our simulation implementation. We implemented DIS algorithm in a simplified simulator, where a round based message passing mechanism with hop delay is provided. At each simulation round, events are scheduled to be processed at each sink and sensor and this procedure may generate new events to be scheduled later. We did not implement the data traffic nor the lower MAC layer. Instead, the bandwidth utilization is computed by the traffic load function (1) and we assume a perfect scheduling mechanism is employed by the underlying MAC layer which achieves the link capacity. All simulations follow a common set of network settings depicted in Table 1.

Our first simulation studies the overall capacity utilization u of the network under a OPT configuration,

Table 1. Simulation Parameters

Network area	1000m × 1000m
Number of sensor ($\|\mathcal{V}\|$)	3000
Number of sink ($\|\mathcal{S}\|$)	Variable
Sensor distribution	Poisson
Sink distribution	Poisson
Wireless Tx/Rx bandwidth (W)	12.8kbps
Wireless Tx/Rx radius (r_0)	50m
Query data rate (b)	20bps
Query radius increase interval	10 simulation rounds
Query radius increase step	10m
Total simulation rounds	400

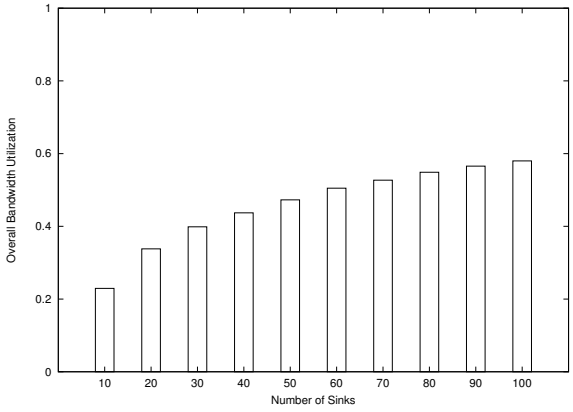


Figure 7. Overall bandwidth utilization.

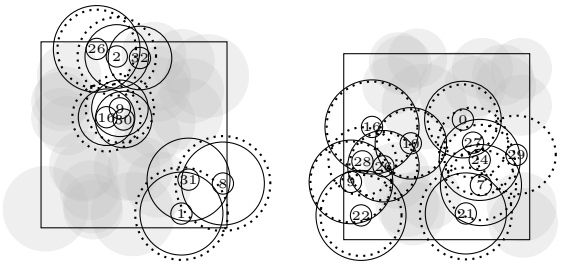
where u is defined as:

$$u = \frac{\sum_{i \in \mathcal{V}} \sum_{j \in \mathcal{S}} \delta(d_{ij}, r_j)}{\frac{nW}{\pi r_0^2 \lambda_0}}$$

Note that the traffic load function δ defined in (1) becomes a function of both d_{ij} and r_j since the sinks may have different query radii.

In order to eliminate random effects, we run the simulation with a hundred different topologies. Figure 7 shows the results. We see that the bandwidth utility grows with the sink number but at a decreasing rate. On the other hand, the utility is quite low especially when less sinks are in the network. This is due to the query model we have employed. Future works will have to consider different query models in order to achieve a higher network utility.

The second simulation focuses on the distributed algorithm. We observe the query radius during a simulation run and show how DIS algorithm copes with sinks joining and leaving the network. So, 30 sinks are initially deployed and they start a query at the beginning of the simulation. Then at simulation round 200, three sinks join the network and start their queries. We record the radii of each sink at each simulation round and compare it with the OPT query radii under the same network topology.



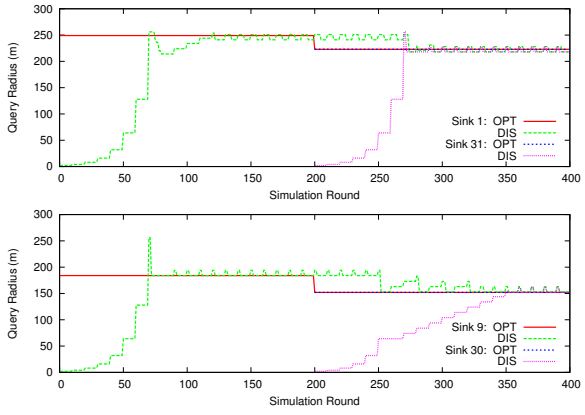
(a) Sink 30, 31, 32 added. (b) Sink 27, 28, 29 removed.

Figure 8. Query radii dynamics.

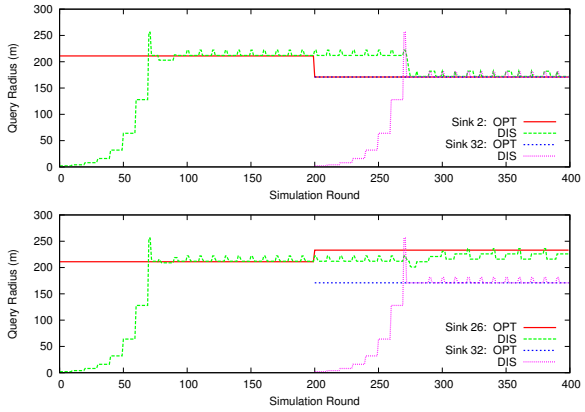
Figure 8(a) shows the OPT radius before and after three sinks are added in the network. Gray plates denote queries not affected by sink addition or removal. Dotted and solid circles denote the affected optimal query radii before and after topology change, respectively. Most of the affected queries shrink in order to share the network capacity with the newcomers, for example query 1 and 8 are forced to shrink by query 31. However, some affected queries enlarge. This may happen when their neighboring queries are forced to shrink. Such behavior is expressed by sink 26 since query 2 is further limited by a new query 32. Similar behaviour was observed when sinks leave the network in Figure 8(b), where most of the impacted sinks experience an enlarged query radius, while a few others have to shrink, *e.g.* sink 21.

Now we analyze how the query radii evolve with time. In Figure 9(a), four pairs of sinks are selected for demonstration: sink 1 with 31, 9 with 30, 2 with 32 and 26 with 32. In each pair, the query of the first (original) sink is affected by the second (newly added) sink. The OPT values are also plotted as references.

The two plots in Figure 9(a) show that the slow start procedure eventually reaches the OPT value. After that, the distributed algorithm will determine an approximation to the OPT value with LOCAL algorithm. The approximation is good since it coincides with the OPT value for most of the cases, *e.g.* for sink 9 and 30. Although the approximation is below the OPT value for sink 1 and 31, the difference between them is small. Thus, based on this approximation, the `increaseRadius` procedure is able to achieve the OPT value very quickly. Besides, we could see that query radius of the original sinks (sink 1 and 9) begin to drop when the new query grows large enough to produce an interaction between them. For sink 1, the interaction takes place when the new query 31 is above the OPT value. As a result, they immediately find the bottleneck between them (also for sink 8, as shown in Figure 8(a)). However, the interaction



(a) Sink 1 and 9 with sink 31 and 30 affecting them, respectively.



(b) Sink 2 and 26 with sink 32 affecting them.

Figure 9. Evolution of query radii decided by DIS algorithm.

happens quite early for sinks 9 and 30. This could be explained by the fact that different sensors emit <saturated> messages at different time when the query is expanding until the actual bottleneck is found. Once the first <saturated> message is received, sink 30 quits the slow start procedure and increases its query linearly, so the time required to achieve the OPT value is significantly increased (from round 250 to 350).

The two plots in Figure 9(b) records the query radii of sink 2, 26 and 32 in Figure 8(a). The higher one shows the interaction between query 2 and 32. It is shown in order to be compared with the lower one, which is more interesting. In this plot sink 26 and 32 have different OPT values because they are not sharing a common bottleneck sensor, thus, not directly interacting with each other. However we see that once the new sink 32 achieves its OPT values, query 26 increases its radius and adjusts it to its own OPT values, which is higher than its original OPT value. This can happen because sink 2 is forced to shrink its query, which eliminates the bottleneck between sink 2 and 26, as shown in Figure 8(a).

6. Related works

The main concern of this paper is to share the bandwidth capacity among the sinks, however the capacity of wireless networks itself deserves separated research efforts and has been studied intensively since the seminal paper [8]. Theoretical results show that under the optimal case, the per-node transport capacity of ad hoc networks scales as $O(1/\sqrt{N})$, if independently and randomly distributed traffic is assumed. However, in typical WSN applications, the data traffic are usually directed to the same destination, making the traffic pattern in WSN quite different from traditional ad hoc networks. In [5], the per-node throughput capacity was shown to scale as $\Theta(1/N)$. We have adopted the result of [5] as the basis of our derivation since it meets our assumptions of the network.

The multi-sink paradigm studied in this paper is gaining more emphasis from the WSN research community. Routing mechanisms have been developed to support efficient data collection to multiple sinks in [9], [10]. Network architecture has been proposed in [11] where mobile phones able to communicate directly with the sensors through multiple wireless interfaces act as sinks. In such networks, it is natural to have multiple, or even a mass of, sinks. Our firemen scenario is one of the potential applications of this paradigm.

Our protocol consists of a congestion control mechanism aiming to provide both reliability and fair capacity sharing to all queries. Traditionally, these three functions are the basis of a transport protocol. However, we emphasize more on bandwidth sharing among sinks in a max-min fair way, which is different from previous works where fairness is either applied to sources (sensors) or neglected. Although we have employed a mechanism similar to the slow start procedure in TCP, there exists an important difference. On detecting a potential congestion state, our protocol shrinks the related queries to a theoretically safe value based on the LOCAL algorithm to avoid the congestion, while TCP-like mechanisms set the new contention window to an empirical value, for example, to the minimum value, half of the current value or the slow start threshold.

Max-min fairness has been studied within different contexts such as in wireless ad hoc networks [12] or wireless mesh networks [13]. Especially, it has been studied for WSNs in [14], but at the MAC layer. Besides, network utilization has been maximized with

max-min fairness in [15]. This last paper is very close to ours in that both achieve a maximum network utilization under bandwidth constrains and provide max-min fairness. However, fairness is based on traffic flows on each sensor and only one sink is considered in [15]. Fair capacity sharing between multiple sinks has never been formalized and studied before, to the best of our knowledge.

Paper [16] also deals with the problem of sharing a WSN among multiple users but it emphasizes more on the efficiency. Algorithms have been proposed to merge the queries from different users to meet the bandwidth constraint of the WSN. This work and ours are complementary since we have dealt with different aspects of the same problem. In fact, it could be interesting to apply fairness constraints to a WSN with merged queries.

7. Conclusion

In this paper, we have dealt with the problem that multiple sinks having to adjust their query ranges according to max-min fair rules in order to meet the capacity constraint of a WSN.

We have proposed a query model based on a disk centered at a sink with variable radius and derived analytically the max-min fair radius for each sink for two-sink case. For general multi-sink case, a distributed algorithm was proposed to enable each sink determining a near-optimal radius most of the time with information provided by sensors under its query coverage.

However, this is only a preliminary research on this problem, future work is necessary to solve the problem thoroughly. First, simulations show that the overall bandwidth is not fully utilized. The query model could be modified to match better the capacity of a region. Besides, the relationship between fairness and bandwidth utilization in WSN with multiple sinks deserves further investigation. Research efforts are needed if sink mobility is considered. Simulation results show that the algorithm sometimes reacts slowly to topology changes, which would hardly be acceptable in mobile sink scenario.

References

[1] D. Culler, D. Estrin, and M. Srivastava, "Overview of sensor networks," *IEEE Computer*, vol. 37, no. 8, pp. 41–49, 2004.

[2] M. B. S. Andreas Savvides, Chih-Chieh Han, "Dynamic fine-grained localization in ad-hoc networks of sensors," in *ACM Mobicom*, 2001.

[3] Y. Ganjali and A. Keshavarzian, "Load balancing in ad hoc networks: Single-path routing vs. multi-path routing," in *IEEE INFOCOM*, 2004.

[4] J. Luo and J.-P. Hubaux, "Joint mobility and routing for lifetime elongation in wireless sensor networks," in *IEEE INFOCOM*, 2005.

[5] D. Marco, E. J. Duarte-Melo, M. Liu, and D. L. Neuhoff, "On the many-to-one transport capacity of a dense wireless sensor network and the compressibility of its data." in *IEEE/ACM IPSN*, 2003.

[6] J.-Y. L. Boudec, "Rate adaptation, congestion control and fairness: a tutorial," in *Technical report.*, 2006. [Online]. Available: http://ica1www.epfl.ch/PS_files/LEB3132.pdf

[7] J. Widmer, R. Denda, and M. Mauve, "A survey on TCP-friendly congestion control," *IEEE Network*, vol. 15, no. 3, pp. 28–37, 2001.

[8] P. Gupta and P. R. Kumar, "The capacity of wireless networks," *IEEE Transactions on Information Theory*, vol. 46, no. 2, pp. 388–404, 2000.

[9] K. il Hwang, J. In, and D. S. Eom, "Distributed dynamic shared tree for minimum energy data aggregation of multiple mobile sinks in wireless sensor networks," in *European Conference on Wireless Sensor Networks (EWSN)*, 2006.

[10] P. Ciciriello, L. Mottola, and G. P. Picco, "Efficient routing from multiple sources to multiple sinks in wireless sensor networks," in *European Conference on Wireless Sensor Networks (EWSN)*, 2007.

[11] C. Chen and J. Ma, "Mobile enabled large scale wireless sensor networks," in *International Conference on Advanced Communication Technology (ICACT)*, 2006.

[12] X. L. Huang and B. Bensaou, "On max-min fairness and scheduling in wireless ad-hoc networks: analytical framework and implementation," in *ACM MobiHoc*, 2001.

[13] B. Aoun and R. Boutaba, "Max-min fair capacity of wireless mesh networks," in *IEEE International Conference on Mobile Ad-hoc and Sensor Systems (MASS)*, 2006.

[14] A. Sridharan and B. Krishnamachari, "Max-min fair collision-free scheduling for wireless sensor networks," in *IEEE International Performance Computing and Communications Conference (IPCCC)*, 2004.

[15] ——, "Maximizing network utilization with max-min fairness in wireless sensor networks," in *International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOpt)*, 2007.

[16] R. Muller and G. Alonso, "Efficient sharing of sensor networks," in *IEEE International Conference on Mobile Ad-hoc and Sensor Systems (MASS)*, 2006.