Math Saves the Forest

Analysis and optimization of message delivery in wireless sensor networks

Peter Korteweg[1], Misja Nuyens[2], Rob Bisseling[3], Tom Coenen[4], Henri van den Esker[5], Bart Frenk[1] Roland de Haan[4], Birgit Heydenreich[6], Remco van der Hofstad[1,7] Jos in 't Panhuis[1], Lieneke Spanjers[8], Maarten van Wieren[7]

Abstract

Wireless sensor networks are decentralised networks consisting of sensors that can detect events and transmit data to neighbouring sensors. Ideally, this data is eventually gathered in a central base station. Wireless sensor networks have many possible applications. For example, they can be used to detect gas leaks in houses or fires in a forest.

In this report, we study data gathering in wireless sensor networks with the objective of minimising the time to send event data to the base station. We focus on sensors with a limited cache and take into account both node and transmission failures. We present two cache strategies and analyse the performance of these strategies for specific networks. For the case without node failures we give the expected arrival time of event data at the base station for both a line and a 2D grid network. For the case with node failures we study the expected arrival time on two-dimensional networks through simulation, as well as the influence of the broadcast range.

Keywords: sensor networks, data gathering, stochastic optimisation, distributed algorithms, random walks, first-passage percolation.

6.1 Introduction

Suppose that you want to design a system to detect fires in a forest. You consider placing sensors that can detect a fire in their neighbourhood. Since

1: Technische Universiteit Eindhoven, 2: Vrije Universiteit Amsterdam, 3: Universiteit Utrecht, 4: Universiteit Twente, 5: Technische Universiteit Delft, 6: Universiteit Maastricht, 7: EURANDOM, 8: CQM

these sensors work on battery power, immediately restrictions arise that make the design of such a system an interesting endeavour. First, transmitting a message to a receiver outside the forest may cost too much energy. In that case, only short-range transmissions are possible. Also, it may not be feasible or too costly to replace the battery of the sensors on a regular basis, so the battery lifetime should be made as long as possible. On the other hand, you want to be sure that the message that there is a fire is transmitted to the receiver outside the forest, and moreover, this should not take too much time.

The question how to design such a forest fire detection system, and control the efficiency of such a system in terms of observing a fire at the base station given possible sensor failures, is an example of the question posed to SWI 2006 by Chess [1]. Chess is a middle-size company providing products and services in the field of electronics, IT-applications, and embedded software. At the moment, Chess considers designing so-called wireless sensor networks for a broad range of applications. We shall describe those networks in more detail further on in this introduction. Apart from detecting forest fires, one could think of detecting gas leaks in neighbourhoods, monitoring the functioning of street lights, as well as using the system for picking up garbage: in many Dutch cities, garbage is collected in large underground bins, and these bins could send a message when they are (almost) full and need to be emptied.

A wireless sensor network is a network that consists of small devices that communicate with each other through radio signals. See Figure 6.1 for an example of such a sensor. Such devices, named sensor nodes, are able to monitor their environment, collect environmental data, process these data and communicate them to other nodes [6, 7].

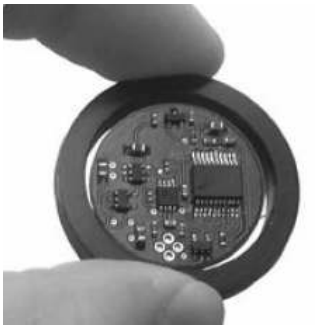

Figure 6.1: A wireless sensor

Sensor networks have several characteristics that distinguish them from wired networks, see [9]. We list the differences with an emphasis on those differences that influence the design of communication algorithms:

- Sensors mainly use *broadcasting* to communicate data. A sensor node that broadcasts, sends data via a radio signal to all sensors in its neighbourhood.

- Sensor networks are *distributed* networks, i.e., they lack a central coordinator. As a consequence, each node has to decide itself what it communicates and when.
- Sensor nodes are *limited in memory and power*.
- Sensor *nodes* are prone to *failure*, i.e., a node may break down and stop operating.
- Wireless *communication* is prone to *failure*.

One of the key research problems in the area of sensor networks is finding efficient communication algorithms. Much research has focused on finding such algorithms for wired networks; see for example the surveys [3] and [5]. However, due to their characteristics, communication algorithms for wired networks do not necessarily provide algorithms for wireless sensor networks. Therefore, in recent years research focused on finding efficient communication algorithms for wireless sensor networks; see [9] for an overview of such algorithms.

In this report, we study a communication problem on a static wireless sensor network, the Sensor Data Gathering Problem (SDGP). In this problem, stations (sensor nodes) in the network provide data that need to be gathered at a base station; the stations are assumed to be static. These data consist of events that occurred in the neighbourhood of a node, e.g. a fire. Stations may communicate messages of events through broadcasting and each message contains information concerning a single event. The objective of the SDGP is to find an efficient algorithm for data gathering at a base station of a wireless static sensor network. Data gathering means that for each event at least one message containing the data should reach the base station. In the literature, there exist several concepts of efficiency. These concepts focus on minimising a function of the completion time of data gathering or maximising a function of the battery lifetime. In this paper, we mainly focus on the objective of minimising the completion time. So, generally speaking the objective is to send messages to the base station as fast as possible.

We emphasise three specific characteristics of our sensor network. First, the network is prone to two types of failure: communication failure and node failure. Second, nodes have a limited memory to store messages, called the *cache*. Due to their limited cache size, sensors should have a *cache strategy*, which determines which message to delete in case of a cache overflow. Third, for design purposes and to limit battery power, sensors are simple devices with a limited set of operations. To communicate their data, sensors use broadcasting. Thus, we assume that sensors cannot use any specific routing information, i.e., sensors are unable to establish point-to-point communication of messages.

Summarising, a communication algorithm should consist of a protocol that decides which messages to broadcast, and of a cache strategy. In this paper, we analyse the performance of several communication algorithms for specific network structures: the 1D grid, the 2D (square) grid and the 2D hexagonal

grid. Since the charm and power of the described sensor networks lies in their simplicity, we focus on communication algorithms that are as simple as possible.

The paper is organised as follows: in Section 6.2, we give a mathematical formulation of the problem. In Section 6.3, we give a mathematical analysis for the case without node failures and unit broadcast radius. First, we analyse the SDGP with unlimited cache size. In this case, there is no need for a cache strategy and message detection by the base station is independent of other messages. We give a probabilistic analysis of the expected number of rounds before an event is detected by the base station. Then, we analyse the SDGP with a cache size of one. In this case, events cannot always be detected by the base station. We give a probabilistic analysis for the case with two events. In Section 6.4, we consider the more general case with node failures and arbitrary broadcast radius. Our results in this section are based on simulations only. In Section 6.5, we summarise the results and give recommendations for designing efficient communication algorithms.

6.2 Problem formulation

We formulate the Sensor Data Gathering Problem as a graph problem. Let $G = (V, A)$ be a directed graph with vertex set V, edge set A, and let $|V| = n$. Also given are a *sink* $s \in V$, a set of events $E = \{1, \ldots, |E|\}$, a set of messages $M = \{1, \ldots, m\}$ for some integer m, an integer cache size $c > 0$, an integer broadcast radius $r > 0$ and probabilities $p > 0$ and $q > 0$, defined below.

The nodes of the graph are stations and the sink is the base station. For each pair of nodes $u, v \in V$, we define the *distance* between u and v, denoted by $d(u, v)$, as the edge cardinality of a shortest path from u to v in G. Given radius r let $N_r(u) = \{v | d(u, v) \leq r\}$ be the neighbourhood of u and let $v \in N_r(u)$ be a neighbour of u. In case $r = 1$, the neighbours of u are those nodes v such that $(u, v) \in A$. We assume that time is discrete, say $\{1, 2, \ldots\}$; a time instance is called a *round*. We assume that sensor nodes have a clock, and that all clocks are synchronised.

Each event $e \in E$ contains data, e.g "There is a fire", a *source node* v_e, i.e., the node where the event was detected, and a *detection time* t_e, i.e. the first time the source node detected the event. Nodes may communicate with each other and if they communicate, they exchange messages. Each message $j \in M$ contains data of a single event e, including source node and detection time. It also contains a timestamp, indicating the round in which the message was sent by its source node. Nodes may use this information to schedule messages. We assume that once the source node of event e detects this event, it creates a message for this event in *each* subsequent round. Note that these messages all have the same detection time, but different timestamps.

Each node has a cache to store messages. We assume the cache consists of a receiver cache of unlimited size and a sender cache of size c. During a round, nodes may communicate with each other through *broadcasting*. A node that

broadcasts sends a copy of each message in its sender cache to all its neighbours. So, if node u broadcasts its sender cache, then each node $v \in N_r(u)$ receives the content of the sender cache of u and stores the information in its receiver cache. A node may broadcast at most once during a round. We assume that node broadcasts do not interfere with each other, hence there is no collision of messages.

A sensor network is prone to two types of failure: communication failure and node failure. We define q as the probability that a broadcast from node u to v during a round is a success, for any $(u, v) \in A$. Moreover, we assume that broadcasts fail independently. In particular, this means that if in a round node u broadcasts to both v and v', then each node has probability q to receive (the same) data.

Since the time scale for the transmission of a message through the network is of a different order than the lifetime of a node, we assume that nodes do not fail during the period that we consider. We call a node *active* if it is operational, i.e., it has not failed, and *inactive* if it has failed. We define p as the probability that a node is active, and assume that nodes are active independently of each other.

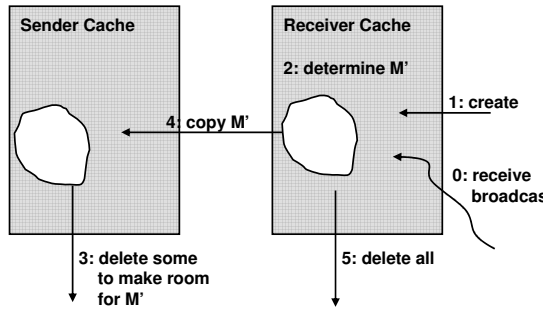

Figure 6.2: The cache strategy

Each node v has a cache strategy. We assume that at the start of a round, all messages received in the previous round are stored in the receiver cache. A node should then update its cache using its cache strategy. A cache update of node v consists of the following consecutive actions, see also Figure 6.2.

1. Create a message for each event with source node v and store this message in the receiver cache.
2. Choose the set of messages M' to be copied from the receiver cache to the sender cache; below we will consider two ways to choose this set.
3. Delete messages from the sender cache such that all messages in M' can be copied to the sender cache.

4. Copy the messages in M' to the sender cache.

5. Delete *all* messages from the receiver cache.

We make the following two assumptions for each cache update. First, after the cache update the sender cache contains at most one message for each event e. If the cache consists of multiple messages for a single event before the update, then the cache only stores the message with the most recent timestamp. Second, messages are only deleted from the sender cache when necessary.

In case of a limited sender cache c, the node must choose which messages to delete from the sender cache and which messages to copy from the receiver cache to the sender cache. The cache strategy should be based only on local information of a node: the current time and the information of the messages in its cache. Hence, the cache strategy is a distributed algorithm.

We consider two different cache strategies based on how messages are deleted from the sender cache in step 3:

- Random Deletion: Messages are randomly deleted from the sender cache;

- Timestamp Deletion: Messages are deleted by decreasing timestamp, i.e., the message with the oldest timestamp is deleted first. Ties are broken arbitrarily, i.e., if two messages have the same timestamp, one of them is deleted according to some arbitrary but fixed rule.

The cache strategy Timestamp Deletion was introduced by Chess [1]. In fact, Chess' cache strategy also deletes too old messages if the cache is not full. However, this does not make sense here since we do not consider optimising the battery lifetime. In Subsection 6.3, we further comment on this when discussing the 2D grid. Furthermore, note that under strategy Random Deletion it is possible that a node has detected an event, but it does not send a message of this event immediately. Finally, if one or more messages from an event reach the base station, we say that the event (data) has been *gathered* by the base station.

The objective of the Sensor Data Gathering Problem is to gather events at the base station of a wireless static sensor network while minimising the completion time of all events. The completion time of an event is the number of rounds needed for one of the messages corresponding to this event to reach the base station. Thus, generally speaking, we are interested in sending messages to the sink as fast as possible. Since the completion time of an event depends on the probabilities p and q, it is a random variable.

6.3 Probabilistic analysis

In this section, we give a probabilistic analysis for the case without node failures, i.e., $p = 1$ throughout this section. In Subsections 6.3 and 6.3, we consider

unlimited cache size ($c = \infty$) for both the 1D grid and the 2D grid. We are interested in the expected completion time of an event, i.e., the expected number of rounds needed to send some message with data of event e to the sink. As the cache size is infinite, no messages have to be deleted from the sender cache. Hence, the completion time of an event is independent of the possible existence of other messages. Thus we may restrict our analysis to considering detection of a single event. Another consequence is that the cache strategies RANDOM DELETION and TIMESTAMP DELETION are identical. Let the random variable T_d be the number of rounds required to gather at the sink a message whose source node is at distance d from the sink.

In the Subsections 6.3 and 6.3, we consider a cache of size one and give a probabilistic analysis in case two events occur.

The 1D grid with unlimited cache size

Given is a 1D grid of sensor nodes $s, 1, \ldots, n$ with base station s; node i is at distance i from s, see Figure 6.3.

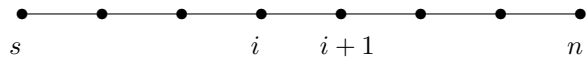

Figure 6.3: 1D grid with base station s

Suppose an event occurs at time 0 at node d. In each round, let X be the node closest to the sink that has received a message of this event. We will also call X the distance of the event to the sink. First, we consider the case $r = 1$, so nodes can only broadcast their cache to their nearest neighbours. In each round, X either moves one step closer to the sink, with probability q, or it remains at the same distance, with probability $1 - q$. In total, X has to travel distance d. This means that T_d is equal to the number of trials needed to obtain d successes, where the probability of a success is q. So, T_d follows a negative binomial distribution with parameters q and d, i.e.,

$$\mathbb{P}(T_d = t) = \binom{t-1}{d-1} q^d (1-q)^{t-d}, \; t = d, d+1, d+2, \ldots \; . \tag{6.1}$$

Note that as a consequence, for a broadcast success probability $q > 0$, the event will be gathered with probability 1. Another consequence of (6.1) is the following.

Corollary 6.3.1. *The expected number of rounds required to gather an event detected at distance d satisfies $\mathbb{E}[T_d] = d/q$.*

Second, consider the situation that a node is able to transmit at a larger range, i.e., $r > 1$. We assume that the success probability of a broadcast

equals q independent of the distance between the nodes. Let Y_r be the effective distance that one particular message gets closer to s in an arbitrary round. The effective distance is the maximum distance over which the communication is successful; hence, Y_r is a random variable.

As the success probability of communication is independent of the radius, the probability to get r steps closer to the sink is q. Similarly, given that this broadcast fails, then the probability to get $r-1$ steps closer to the sink is q. Continuing this argument, we arrive at:

$$\begin{aligned} \mathbb{P}(Y_r = k) &= (1-q)^{r-k}q, \qquad k = 1, 2, \ldots, r, \\ \mathbb{P}(Y_r = 0) &= (1-q)^r. \end{aligned} \tag{6.2}$$

Using (6.2), we are able to find the expected value of Y_r. First we write

$$\begin{aligned} \mathbb{E}[Y_r] &= \sum_{k=0}^{r} k\mathbb{P}(Y_r = k) = \sum_{k=1}^{r} k(1-q)^{r-k}q = q\sum_{i=0}^{r-1}(r-i)(1-q)^i \\ &= qr\frac{1-(1-q)^r}{1-(1-q)} - q(1-q)\sum_{i=1}^{r-1} i(1-q)^{i-1}. \end{aligned} \tag{6.3}$$

To evaluate the sum in (6.2), we write

$$\begin{aligned} \sum_{i=1}^{r-1} i(1-q)^{i-1} &= -\frac{d}{dq}\sum_{i=1}^{r-1}(1-q)^i = -\frac{d}{dq}\frac{(1-q)-(1-q)^r}{q} \\ &= \frac{1}{q^2} - \frac{(1-q)^r}{q^2} - \frac{r(1-q)^{r-1}}{q}. \end{aligned}$$

Plugging this into (6.3), we get

$$\begin{aligned} \mathbb{E}[Y_r] &= r - r(1-q)^r - \frac{1-q}{q} + \frac{(1-q)^{r+1}}{q} + r(1-q)^r \\ &= r + 1 + \frac{(1-q)^{r+1}-1}{q}. \end{aligned}$$

Note that for $r = 1$, we find $\mathbb{E}[Y_1] = q$, which corresponds to Corollary 6.3.1.

However, if $r > 1$, then a message can be overtaken by messages with a later timestamp. Hence, in this case, Y_r is a lower bound on the effective distance that one particular message gets closer to s in an arbitrary round.

In Figure 6.4, we have plotted $\mathbb{E}[Y_r]$ as a function of q for several choices of r. The figure confirms what is intuitively obvious: for broadcast radius $r > 1$, the effective number of steps is much larger than in case $r = 1$. In fact, for $r = 1$ the curve is linear, but for $r > 1$ the curves are larger than the linear curves $f(q) = rq$. The reason for this is that if broadcast to a node at distance r fails, there is still a positive probability that a broadcast to a node of distance less than r is successful, or that a younger message is overtaking the message closest to the base station.

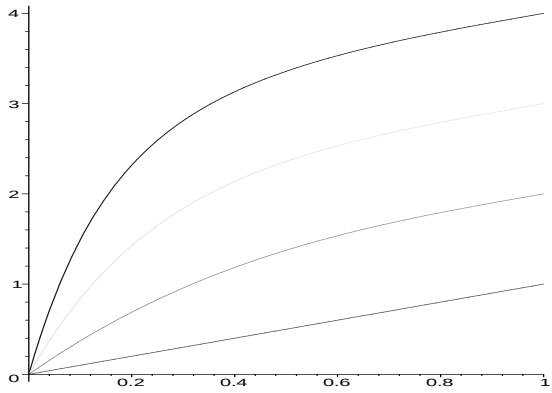

Figure 6.4: $\mathbb{E}[Y_r]$ as a function of q for $r = 1, 2, 3, 4$ (from bottom to top)

The 2D grid with unlimited cache size

Given is a 2-dimensional finite grid of size $\sqrt{n} \times \sqrt{n}$ for some integer $\sqrt{n}$ and with base station s located at one of the corners. Note that on the grid the distance of a node from the base station is at most $2\sqrt{n}$, see Figure 6.5.

Since the probability that an event is gathered is equal to 1 on a 1D grid, as we have seen in the previous section, it is also equal to 1 on a 2D grid. Therefore, we turn to the analysis of $\mathbb{E}[T_d]$, the expected time that is needed to gather an event whose source node, say v_d, is at distance d from the sink. Let X_a be a random variable indicating the number of rounds needed in order to communicate successfully via the (directed) edge a. Clearly, the variables X_a are independently and identically distributed following a geometric distribution with success probability q for all edges a. For any path Φ that connects the sensor to the base station, let T_Φ be the random variable that indicates the time needed to successfully communicate the event via path Φ. Finally, let ϕ be a shortest path from node v_d to the base station, where $a \in \phi$ means that edge a is part of the path ϕ. Then

$$\mathbb{E}[T_d] = \mathbb{E}[\min_{\Phi} T_\Phi] \leq \min_{\Phi} \mathbb{E}[T_\Phi] = \mathbb{E}[T_\phi] = \mathbb{E}[\sum_{a \in \phi} X_a] = \sum_{a \in \phi} \mathbb{E}[X_a] = \frac{d}{q}.$$

Note that the upper bound d/q corresponds with Corollary 6.3.1 (when $r = 1$).

The remainder of this section is devoted to so-called first-passage percolation. The theory of first-passage percolation examines how T_d behaves depending on the position of the sensor with respect to the base station. That is, it examines the behaviour of sensors dependent on whether they are for example situated on the same grid-line as the base station or whether their position is diagonal with respect to the base station. Note that this position influences the number of shortest paths over which a message can be communicated from its source node to the base station.

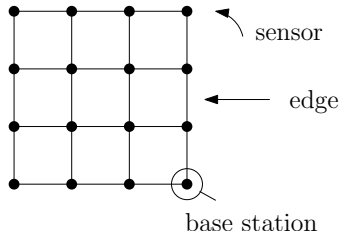

Figure 6.5: The 2D grid

First-passage percolation with geometric distributions has attracted much attention in the literature (see e.g. [4, 8]). An important result, the so-called shape-theorem, describes the shape of the set of points on the grid that can be reached through communication starting from a fixed source sensor within a certain time. The following theorem is implied by the shape-theorem and we use techniques from first-passage percolation to prove it. The theorem provides an upper bound on the probability that the time needed to communicate the event via a specific shortest path ϕ of length d is εd more than the expected time d/q for some positive constant ε.

Theorem 6.3.2. *The probability that T_ϕ exceeds its expectation d/q decreases exponentially with the excess time:*

$$\mathbb{P}(T_d \geq \frac{d}{q} + \varepsilon d) \leq e^{-d\frac{\varepsilon^2 q^2}{2(1-q)}}.$$

Proof. Consider $\mathbb{P}(T_d \geq dk)$ for some positive constant k and let ϕ be a shortest path from node v_d to the base station. Then for all $t \geq 0$ the following is true:

$$\begin{aligned}\mathbb{P}(T_d \geq dk) \leq \mathbb{P}(T_\phi \geq dk) &= \mathbb{P}(\sum_{a\in\phi} X_a \geq dk) \\ &= \mathbb{P}(e^{t\sum_{a\in\phi} X_a} \geq e^{tdk}) \leq e^{-tdk}\mathbb{E}[e^{t\sum_{a\in\phi} X_a}].\end{aligned}$$

The last inequality follows from applying the Markov inequality. Since this holds for all $t \geq 0$, we have

$$\mathbb{P}(T_d \geq dk) \leq \min_{t\geq 0} \exp(-dtk + \log \mathbb{E}[e^{t\sum X_a}]) = \min_{t\geq 0}(\exp(-tk + \log \mathbb{E}[e^{tX_1}]))^d,$$

where X_1 is equal to one of the random variables X_a for some edge a on the path ϕ. Hence, we can write

$$\mathbb{P}(T_d \geq dk) \leq e^{-dI(k)},$$

where

$$I(k) = \sup_{t\geq 0}\{kt - \log \mathbb{E}[e^{tX_1}]\} = \sup_{t\geq 0}\{kt - \log \frac{qe^t}{1-(1-q)e^t}\}$$
$$= \sup_{t\geq 0}\{(k-1)t - \log q + \log(1-(1-q)e^t)\}.$$

Calculus yields

$$I(k) = (k-1)(\log(k-1) - \log k - \log(1-q)) - \log q + \log \frac{1}{k}$$
$$= (k-1)\log(\frac{k-1}{1-q}) - k\log k - \log q.$$

Setting $k := \frac{1}{q} + \varepsilon$, we get:

$$\mathbb{P}(T_d \geq d(\frac{1}{q} + \varepsilon)) \leq e^{-dI(\frac{1}{q}+\varepsilon)}.$$

Since $I(\frac{1}{q}) = I'(\frac{1}{q}) = 0$, using the Taylor expansion yields

$$I(\frac{1}{q} + \varepsilon) = I(\frac{1}{q}) + \varepsilon I'(\frac{1}{q}) + \frac{\varepsilon^2}{2}I''(\frac{1}{q}) + o(\varepsilon^2) = \frac{\varepsilon^2}{2}I''(\frac{1}{q}) + o(\varepsilon^2).$$

Finally, calculating

$$I''(\frac{1}{q}) = \frac{q^2}{1-q},$$

completes the proof. □

To illustrate Theorem 6.3.2, Figure 6.6 shows this upper bound for the situation where $\sqrt{n} = 101, d = 200$ (the worst case scenario) and $q = 0.95$.

It is clear that for the expected number of steps needed, which is around 210, the upper bound does not provide much information. However, for the situation with only ten steps more, Theorem 6.3.2 provides strong information: the probability that the message needs more than 220 steps to reach the base station is already below 1%.

Let us emphasise that the given bound only takes the communication via one path into account. In reality, there are multiple paths that can be used, hence it is likely that the expected completion time will be even shorter.

For the example shown, we can also conclude that if the cache strategy would delete messages whose timestamp is at least 220 rounds old then the probability that the first message created for this event does not reach the base station is less than 1 percent. Such a strategy would assure that messages are not kept longer than necessary in the cache, and decreases the amount of old messages circulating in the network. This is beneficial for the lifetime of the batteries in the sensor.

We conclude this subsection with the following remark. In this subsection, we have assumed that sensors do not fail, i.e., $p = 1$. This assumption is not

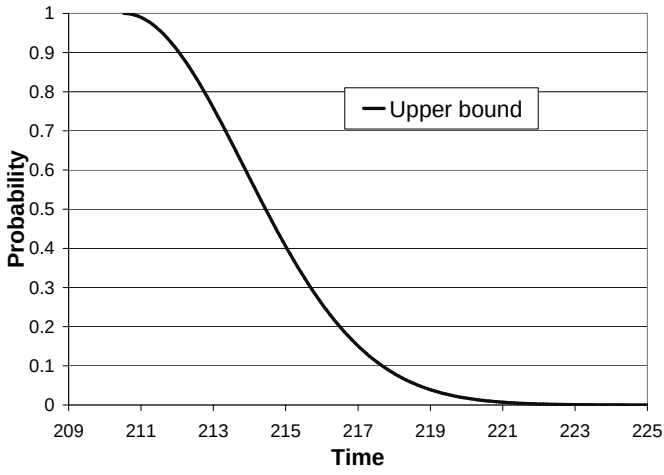

Figure 6.6: The upper bound of Theorem 6.3.2 for $\sqrt{n} = 101, d = 200$ and $q = 0.95$.

very restrictive in the 2-dimensional case, as from an arbitrary sensor in the network, there exist multiple paths toward the base station. So, if one path is not available, there may be many other available candidates. Hence, unlike in the one dimensional case, for p not too far from 1, the probability that the base station can be reached from the sensor by at least one path is close to 1 as well. Of course, it would be an interesting problem to quantify these "not too far from 1" and "close to 1".

The 1D grid with cache size one and two events

We again consider the model on the 1D grid, but this time we assume that $c = 1$ for each node. Given is a 1D grid of sensor nodes $s, 1, \ldots, n$ with base station s; node i is at distance i from s. We assume that the broadcast radius r is 1. We are interested in the probability that if two events occur, both events are gathered. We compare this probability for the cache strategies Random Deletion and Timestamp Deletion.

We assume there are two events 1 and 2 with source nodes v_1 and v_2, respectively, and detection times t_1 and t_2. Without loss of generality we assume that v_1 is closer to the sink than v_2. Figure 6.7 illustrates the situation. First, we consider Timestamp Deletion. In this case, from all the messages that a node receives, it will only send the one with the youngest timestamp, i.e., the message that has been in the system the shortest time. Let A_i be the event that the ith event is gathered. We are interested in calculating the

probabilities $\mathbb{P}(A_1)$ and $\mathbb{P}(A_2)$.

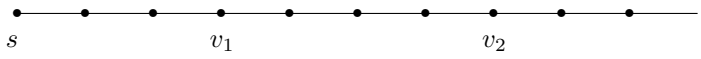

Figure 6.7: *1D grid with events 1 and 2*

The probability $\mathbb{P}(A_1)$ is easily determined. Since messages corresponding to event 1 are sent every round after event 1 has been detected, there will always be a message from this event in the sender cache of v_1. Hence, the probability that a single message reaches the sink is at least q^{v_1}. This implies that the probability that none of the messages sent by v_1 reach the sink is upper bounded by $\lim_{n\to\infty}(1-q^{v_1})^n = 0$. Hence, $\mathbb{P}(A_1) = 1$.

Now we consider $\mathbb{P}(A_2)$. We claim that if $t_1 \leq t_2$, then $\mathbb{P}(A_2) = 0$. Indeed, any message created at v_2 must eventually pass vertex v_1. However, since $t_1 \leq t_2$, this vertex is already busy sending messages of its own event. As the timestamp of these messages is always younger than the timestamp of messages from event 2, messages from event 2 can never pass v_1, and thus never reach the sink. We conclude that if we want both events to be gathered at the sink, then the cache strategy with Timestamp Deletion is bad one.

Next, suppose that in Timestamp Deletion we have $t_2 < t_1$. For each round, let the random variable X be the position of the message from event 2 that is closest to the sink at time t_1. Set for notational convenience $\tau = t_1 - t_2$. If $0 < k \leq v_2$, then $\mathbb{P}(X = k)$ is the probability of $v_2 - k$ successes in τ trials with success probability q. Hence, X is binomially distributed with parameters τ and q. Furthermore, $\mathbb{P}(X = 0)$ is the probability of at least v_2 successes in τ trials. Thus,

$$\mathbb{P}(X = k) = \binom{\tau}{v_2 - k} q^{v_2-k}(1-q)^{\tau-v_2+k}, \quad \text{for } 0 < k \leq v_2, \tag{6.4}$$

$$\mathbb{P}(X = 0) = 1 - \sum_{i=0}^{v_2-1} \binom{\tau}{i} q^i (1-q)^{\tau-i}. \tag{6.5}$$

Now we condition on the value of X. Since the strategy with Timestamp Deletion implies that $\mathbb{P}(A_2 \mid X = k) = 0$ for all $k \geq v_1$, we have

$$\begin{aligned} \mathbb{P}(A_2) &= \sum_{k=0}^{\infty} \mathbb{P}(A_2 \mid X = k)\mathbb{P}(X = k) \\ &= \sum_{k=0}^{v_1-1} \mathbb{P}(A_2 \mid X = k)\mathbb{P}(X = k). \end{aligned} \tag{6.6}$$

To find these conditional probabilities, we consider the following situation. Let i and j be the position closest to the sink of the messages from v_1 and v_2

respectively. Let $p(i,j)$ be the probability that from this situation a message from v_2 reaches the sink. One round later, these positions are $i-1$ and $j-1$ with probability q^2, and in that case the desired probability is $p(i-1,j-1)$. By also considering the other possibilities for the situation one round later, we find that $p(i,j)$ satisfies the recurrence relation:

$$p(i,j) = q^2 p(i-1,j-1) + q(1-q)p(i,j-1) + q(1-q)p(i-1,j) + (1-q)^2 p(i,j). \tag{6.7}$$

Since messages from v_1 have priority over those from v_2, the boundary conditions are $p(i,j) = 0$ if $i \leq j$, and $p(i,0) = 1$ for all $i > 0$. Finally, observe that $\mathbb{P}(A_2 \mid X = k) = p(v_1, k)$. Hence, we can calculate (6.6) by solving the recurrence relation (6.7). Unfortunately, there is no easy closed-form solution of this recurrence relation, so that it is only useful for numerical purposes. In this report, we will not explore this numerical path.

Now we consider Random Deletion. To simplify the analysis, we assume that $q = 1$. The case $q < 1$ will be studied via simulations in Section 6.4. If $v_2 - v_1 > t_1 - t_2$, then $\mathbb{P}(A_1) = 1$. We are therefore interested in finding $\mathbb{P}(A_2)$. Let t be the first round such that the sender caches of two adjacent nodes contain different messages. This situation is illustrated in Figure 6.8; here messages from v_1 are denoted by a circle, and those from v_2 by a square.

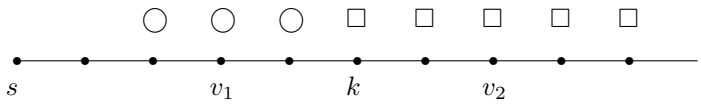

Figure 6.8: The first round that some node k contains a message of event 2 and node $k-1$ contains a message of event 1.

Observe that the model from time t onward resembles a symmetric random walk (RW). Indeed, as a consequence of the Random Deletion cache strategy, one round later, both nodes $k-1$ and k are a circle, or a square, both with probability 1/2, see also Table 6.1. So, with probability 1/4, the front of the squares moves forward to $k-1$, with probability 1/4 it stays in k, and with probability 1/4 it moves back to $k+1$. The only difference with the RW model is the fourth option: a square moves to $k-1$, and a circle moves to k. Although we have not managed to prove it, this fourth option seems to be no worse than the situation in which the front stays at k.

We may conclude that the front of squares reaches the sink in a time comparable to that of a RW with step size distribution

$$\mathbb{P}(X = 1) = \mathbb{P}(X = -1) = \mathbb{P}(X = 0)/2 = 1/4. \tag{6.8}$$

time t		time $t+1$		
$k-1$	k	$k-1$	k	situation
◯	□	◯	□	non-moving front
		◯	◯	progressing front for message 1
		□	□	progressing front for message 2
		□	◯	mixing front

Table 6.1: The four options for the evolution of the front.

So, results for this RW may give us (upper) bounds for the behaviour of the stream of squares. To describe the behaviour of this RW, we first quote two propositions about simple symmetric RW's, i.e., RW's that move one to the right or one to the left, both with probability 1/2. These can be found in Chapter XIV of [2].

Proposition 6.3.3. *Consider a simple symmetric RW. Let $p(x,n)$ denote the probability that starting at $x \in \{s,1,\ldots,n\}$, the message reaches s before it reaches n. Then $p(x,n) = 1 - x/n$.*

This proposition is also known as the Gambler's ruin probability. Since the step size of our RW may be 0, it is not a simple symmetric RW. However, since the step-size distribution is symmetric and concentrated on $\{-1,0,1\}$, the effective steps do form a RW. Hence, the result of the proposition holds for our RW as well.

The proposition tells us that every time the front is in position $v_2 - 1$, the probability that the message from v_2 will be gathered is $1/v_2$. But every round there will be a message originating from v_2 in $v_2 - 1$ with probability at least 1/2. So, with probability 1 there will be infinitely many trials with succes probability at least $1/v_2$ to gather the event detected by v_2. We conclude that the probability that the event from v_2 is gathered is 1, for every node v_2. The following proposition is about a RW with a reflecting barrier. This means that if the message moves from $n-1$ to n, the next round it moves back to $n-1$.

Proposition 6.3.4. *Consider a simple symmetric RW with a reflecting barrier in n. Let $\tau(x,n)$ denote the expected time to reach s starting at x. Then $\tau(x,n) = x(2n-x)$. So, the expected time to reach s from position n is n^2.*

For the RW given by (6.8), the number of steps before a non-zero step is made is geometrically distributed with parameter 1/2, so it has expectation 2. As a consequence, for this RW we have $\tau(x,n) = 2x(2n-x)$.

If we consider the messages with source node v_2, then we can view node v_2 as a reflecting barrier. Indeed, consider the situation that all nodes to the left of v_2 are circles. Since v_2 is the source of the square messages, it has a square in its sender cache with probability 1/2. Hence, with probability 1/2 it sends a square to node $v_2 - 1$, so that the next round, $v_2 - 1$ is a square with

probability 1/2. This is exactly the behaviour of the RW (6.8) with a reflecting barrier in v_2.

Until time t, messages from v_2 move to the sink independent of messages from v_1. From time t onward, messages from v_2 move towards the sink at a speed comparable to a RW with step size distribution (6.8). Hence, by Proposition 6.3.4, the expected time to reach the base station is roughly $2v_2^2$.

For the case $v_2 - v_1 < t_1 - t_2$, we can use similar arguments to find that $\mathbb{P}(A_1) = \mathbb{P}(A_2) = 1$, and to find the time to gather the event at v_1. Finally, we should remark that in our analysis we have ignored that the event at node v_1 may form an extra obstacle for messages from v_2: since v_1 always has a circle in its receiver cache, it is (slightly) more difficult for the squares to pass this node than to pass a normal node.

The 2D grid with cache size one and two events

Given is a 2-dimensional grid of size $\sqrt{n}\times\sqrt{n}$ for some integer $\sqrt{n}$ and with base station s located at one of the corners, see also Figure 6.5. We assume that the size of the sender cache, c, is 1 for each node, and that the broadcast radius r, is 1 as well. We begin by making the observation that in case there is only one fire, the behaviour of the system is equivalent to first-passage percolation, see also Section 6.3. We are interested in the probability that if two events occur, both events are gathered. We consider this probability for the cache strategy Timestamp Deletion. The notation is the same as in the previous section, so two events are detected at nodes v_1 and v_2, respectively, and without loss of generality we assume that v_1 is closer to the sink than v_2.

Let $\Delta := v_2 - v_1 + t_2 - t_1$. So Δ can be viewed as the time difference between the first arrival at the base station of messages sent from v_1 and v_2, if both messages are sent independently ($c \geq 2$). From the observations in subsection 6.3 it follows that $\mathbb{P}(A_1) = 1$. For $\mathbb{P}(A_2)$ we consider the case $q = 1$. In this case, the difference in distance to the origin between the two initial points fully determines whether message 1 will reach the origin. This leads to the following theorem:

Theorem 6.3.5. *If $q = 1$, then $\mathbb{P}(A_1) = 1$ and*

$$\mathbb{P}(A_2) = \begin{cases} 1 & \text{if } \Delta < 0 \\ 0 & \text{if } \Delta \geq 3. \end{cases}$$

Proof. As v_1 is closer to the sink than v_2, a message of v_1 that is sent over a shortest $v_1 - s$ path is always forwarded towards the sink, because for any node u on this path its timestamp is later than that of a message from v_2 at this node u. Hence, $\mathbb{P}(A_1) = 1$. If $\Delta < 0$, then the first message sent from v_2 arrives at each node of a shortest $v_2 - s$ path before a message from v_1 can reach this node. Hence, a message from v_2 arrives at s before a message from v_1, thus $\mathbb{P}(A_2) = 1$.

Consider a message from v_2, sent at time t_2', that reaches a neighbour of s in round t. Since s is in the corner of the grid, the distance between neighbours

of s is at most 2. Hence, for all $\Delta \geq 3$, there exists a message from v_1, sent at time $t'_1 > t'_2$, that reaches the same neighbour of s in round t. Since messages from v_1 have a timestamp later than those of v_2 for every neighbour of s, no message sent from v_2 reaches s. If $\Delta = 1$ or $\Delta = 2$, then the probability depends on the position of v_1 relative to v_2. □

From this theorem we may derive that using cache strategy TIMESTAMP DELETION both events are gathered when the first message of the event which was detected furthest (v_2) could have reached the base station before the first message of v_1. On the other hand, if the message from v_2 could only have reached the sink at least 3 rounds later, it never reaches the sink. As in the case of the 1D grid, this demonstrates that TIMESTAMP DELETION is not a particular good cache strategy if we wish to detect all events.

6.4 Simulations

In this section, we give simulation results for the case with node failures, i.e., we assume $p \leq 1$ throughout this section. In the first paragraph, we consider the problem on a 1D grid when there is a cache size of 1 and there are multiple events. We are interested in the probability that events are gathered under the cache strategy RANDOM DELETION. In the second subsection, we consider the same problem on a 2D square grid and a 2D hexagonal grid.

The 1D grid with cache sizes of one and multiple events

Given is a 1D grid of sensor nodes $s, 1, \ldots, n$ with base station s; node i is at distance i from s. We are interested in the influence of q and r on the message completion times for the cache strategy RANDOM DELETION when there are multiple events. To this end, we have developed a simulation to analyse these completion times for several arbitrarily generated events.

Given are four events $1, 2, 3$, and 4, such that event j is detected at time 0 by node v_j; the distance of v_j to the sink is $30j$. We assume that nodes do not fail, i.e., $p = 1$.

First consider the case $r = 1$, where each sensor can only broadcast to adjacent nodes. The left picture of Figure 6.9 depicts the outcome of a single simulation run for this case. Each vertical 1D grid in the picture represents the message each sensor in the 1D grid transmits at time t. For instance, until time $t = 20$ each message is broadcast through the 1D grid without any problems; after time $t = 20$, the messages of the second event, coloured black, are blocked by a message of the first event, coloured light gray. When we say a message is blocked we mean that it is not sent further towards the sink. From the figure, we see directly that using the cache strategy RANDOM DELETION results in a poor performance of the completion times of the events 2, 3 and 4, whereas messages of event 1 reach the sink without any delay. It seems that event 1 blocks the message of the other events.

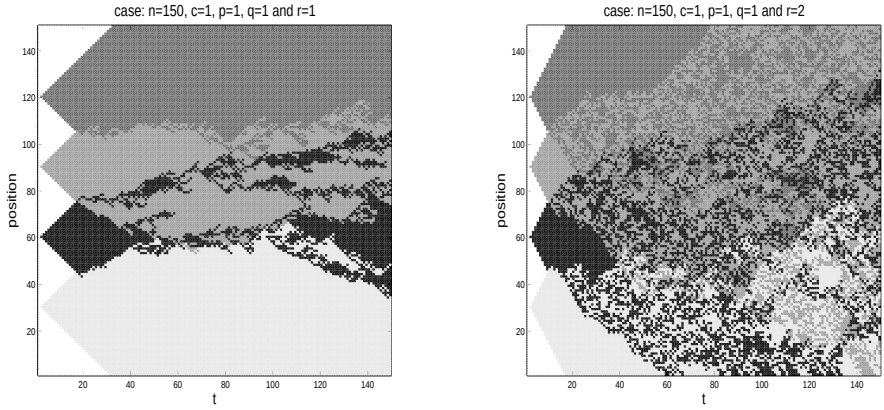

Figure 6.9: Simulation run for the four events starting at $t = 0$ for $r = 1$ (left) and $r = 2$ (right). Each event generates its own unique message, identified by a unique colour. If a sensor did not broadcast any message, then the colour is white. The horizontal axis gives the time (in rounds).

This image changes drastically when we consider a larger broadcast radius, namely $r = 2$. The outcome of a single run is presented in the right picture of Figure 6.9. In this case, the messages become more mixed and as a result also messages from events 2 and 3 reach the sink, within 140 rounds. In particular, we can see that some message of the second event overtakes messages of the first event.

An overview of these observations is plotted in Figure 6.10, which is based on 1000 simulation runs. Note that the horizontal axis in the two figures differs. In the upper figure, representing the situation with $r = 1$, we see that the completion time of event 2 is in general quite large. This gets even worse if we consider the situation of $q = 0.95$, i.e., the case where communication is prone to failures. Although the completion time of event 1 is hardly affected, the completion time of event 2 increases substantially due to the broadcast failures. The lower picture of Figure 6.10, representing the situation with $r = 2$, demonstrates the strongly decreased completion times of event 2. Note also that the impact of broadcast failures (i.e., the $q = 0.95$ case) is smaller than in the $r = 1$ case.

However, in practice it is not always possible to extend the broadcast range to increase the performance. Therefore, another approach would be to change the cache strategy such that messages become more intertwined and in this way keep completion times small. The idea is that a sensor refrains from transmitting the same message all the time and this is formulated in the alternative cache strategy Random Deletion+.

- Random Deletion+: Messages are randomly deleted from the sender cache. The selection of messages to be copied from the receiver cache to the sender

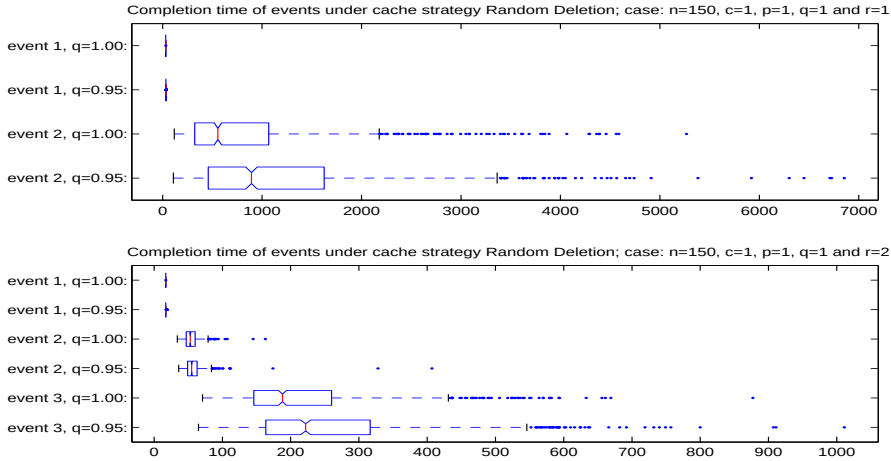

Figure 6.10: The completion time of messages in the form of a box plot. The box has lines at the lower quartile, median, and upper quartile values. The whiskers are lines extending from each end of the box to show the extent of the rest of the data. Outliers are data with values beyond the ends of the whiskers. The whisker extends to the most extreme data value within 1.5· IQR of the box, where IQR is the width of the interval that contains the middle 50 % of the data. The horizontal axis gives the time (in rounds).

cache is as follows: if the receiver cache contains messages related to an event whose data was broadcast by the same node in the last round, then these messages get low priority: they can only be copied to the sender cache if all other messages are copied as well.

Note that this strategy does not require extra memory. In Figure 6.11, we depict single simulation runs for $r = 1$ (left) and $r = 2$ (right). For both cases we immediately note an improvement compared to the original strategy Random Deletion. The messages of event 1 and 2 no longer block each other and therefore messages of both events travel to the sink without any delay. Unfortunately, it seems that messages of event 4 are still blocked by the messages of the other events.

The results for the completion times over 1000 simulation runs are plotted in Figure 6.12. Note that here the range is much smaller than in Figure 6.11. Also here, the top and bottom ranges differ. The completion times under strategy Random Deletion+ are clearly much smaller than under Random Deletion. The effect of reception failures on the completion times is also negligible.

Thus, using the alternative cache strategy Random Deletion+, we could improve the completion times of the messages. Unfortunately, if there are more than three events then the completion time of all events degenerates, as in the

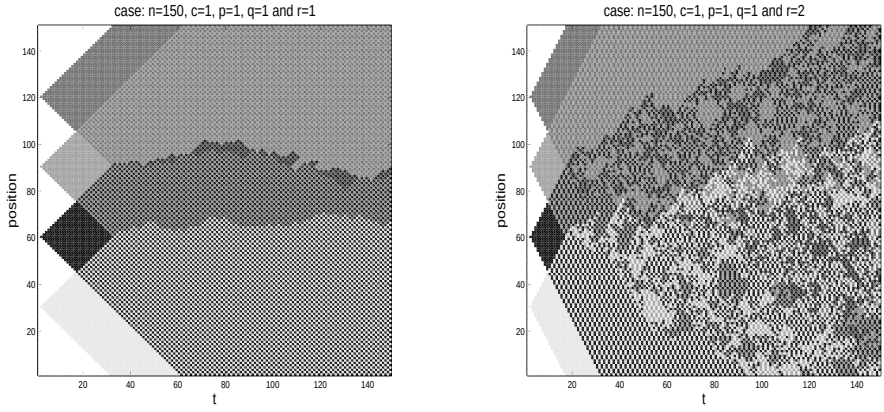

Figure 6.11: Simulations for the cache strategy Random Deletion+. See Figure 6.9 for the interpretation.

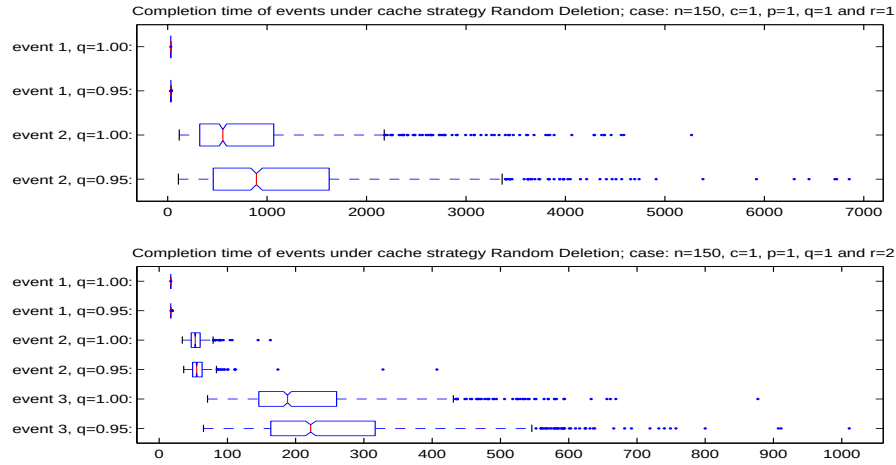

Figure 6.12: See Figure 6.10 for the interpretation, here we use cache strategy Random Deletion+.

case with two events under cache strategy Random Deletion. This could of course be compensated by changing the strategy Random Deletion+ to a more elaborate one, but that would result in a more difficult cache strategy, which might conflict with the aim to keep the strategy as simple as possible.

2D grids with node failures

In the 2D simulations, we consider the area $[0, 100] \times [0, 100]$ covered by sensors with broadcast radius $r = 1$ located at the points of a regular grid. The sensor

placed at $(0,0)$ functions as the base station. We study the data gathering problem for two different grids:

- a square grid with $101 \times 101 = 10201$ sensors numbered $(i,j), 0 \leq i,j \leq 100$.
- a hexagonal grid, where the sensors are located at those points $i(1,0) + j(0.5, 0.5\sqrt{3})$, with i,j integer, that fall within the area. The total number of sensors is 11658.

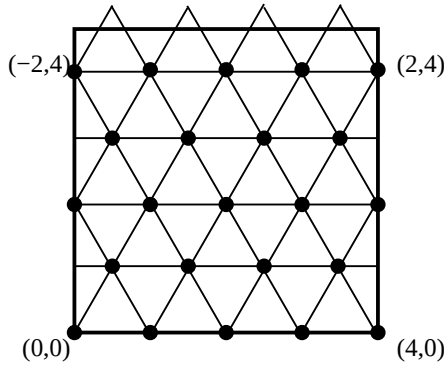

Figure 6.13: Hexagonal grid with 23 sensors covering an area $[0,4] \times [0,4]$. The values (i,j) shown are the coordinates of the corners in the hexagonal system. The longest distance to the base station $(0,0)$ is 6.

The motivation for considering a hexagonal grid, see Figure 6.13, is that each sensor has six neighbours instead of the four in the square grid. We expect this to increase the robustness of the whole system. Furthermore, in the hexagonal grid the distance from the farthest point to the base station is reduced from 200 to 167, thus speeding up the detection of events. A small disadvantage of the hexagonal grid is that more sensors are needed to cover the same area, namely a factor $2/\sqrt{3} \approx 1.16$ more.

In this set of simulations, we study the interference of the messages from the two events. We create two simultaneous events at time $t = 0$, at randomly chosen sensors, and set the cache size $c = 1$. We have run the program 1000 times, each time with a different random number seed. Table 6.2 presents the average number of steps needed to detect the first and second event, for both grids, in case the two events were indeed detected. For $p \leq 0.95$, some runs did not detect any fire. The reason is that failing sensors may cause the graph corresponding to the grid to become disconnected. Furthermore, if an event occurs at a sensor not connected to the base station in the graph, then this event cannot be detected.

The cache strategy we use here is a third variant of Random Deletion, which we call Random Deletion++: the contents of the sender and receive

p	q	Square grid			Hexagonal grid		
		runs	steps 1	steps 2	runs	steps 1	steps 2
1.00	1.00	1000	77.0	141.7	1000	66.1	115.7
	0.95	1000	77.8	146.6	1000	66.2	117.6
0.95	1.00	998	76.2	147.9	996	66.8	117.6
	0.95	998	76.0	148.9	999	67.8	118.5
0.80	1.00	908	80.4	173.1	961	67.0	130.6
	0.95	926	78.4	183.5	940	69.2	135.3

Table 6.2: The number of runs that gathered both events for the strategy RANDOM DELETION++, and the average number of steps needed to gather the first and second event. In total, there were 1000 runs.

cache are merged, duplicates are removed, and messages are randomly deleted until c messages are left. These are then stored in the sender cache. This strategy treats all locally known messages equally (after removal of duplicates), and is not biased towards deleting messages from the sender cache.

The results of Table 6.2 show that the hexagonal grid leads to faster gathering for both events. In particular, the event farthest away from the base station is detected earlier, and its detection time is less affected by failing sensors or failing communications. Note that the ratio of the average gathering times for event 1 corresponds to the ratio of the number of nodes in both networks. A surprising finding is that on the square grid, failing sensors sometimes seem to speed up the detection of the first event, which may be due to less interference from messages for the second event. This is a mixed blessing, as indeed the second event is detected much later.

6.5 Conclusions and recommendations

The main characteristic of a sensor is its simplicity: a sensor has limited processing capabilities, limited power and a limited cache memory. Our objective was to analyse *simple* cache strategies for data gathering in a sensor network. Hence, they should take into account cache constraints, and not use routing information. Our analysis, which consists of an exact analysis based on probability theory and a heuristic analysis through simulation, demonstrates that there exist simple decentralised strategies allowing sensors to gather data efficiently and robust.

We have analysed two strategies: TIMESTAMP DELETION and RANDOM DELETION. In Section 6.3, we have shown that the simplest strategy, TIMESTAMP DELETION, is clearly inferior to RANDOM DELETION, if the number of events is larger than the cache size. Furthermore, simulations in Section 6.4 suggest that the more complicated strategy, RANDOM DELETION+, increases the performance even more. The decision on what level of complexity is allowed

may depend on the application at hand, and should be made by the designers of the system.

A second parameter of interest is the probability that a broadcast fails, q. We have studied how the expected completion time, i.e., the time to gather an event at the base station, depends on q, and the distance from the event to the base station. Different kinds of applications will put different demands on this completion time. For example, forest fires need to be detected immediately, while noise measurements at airports are allowed to come in days later. Hence, per application the system designer should check what values of q are allowed, and what should be done to make sure that q falls within that range.

A third point to consider is the influence of the broadcast range, r. Obviously, the larger the broadcast range, the better. However, due to restrictions on the battery power, only a limited broadcast range may be feasible. The calculations in Section 6.3 reveal the effective step size per round as a function of the broadcast range (and the failure probability q). These results are illustrated and complemented by the simulations in Section 6.4, which show the effect of the broadcast range on the gathering time of events. Again, the demands on the speed by which messages travel through the network should determine how much should be invested in increasing the broadcast range.

Finally, we have considered the influence of the layout of the sensor network in two dimensions. The simulations in Section 6.4 show that a 2D hexagonal layout of sensors is superior to a 2D square layout, both in terms of detection speed and robustness.

We conclude that the performance of a sensor network depends on many parameters. We have tried to describe this performance by analysing some examples of networks. Using this analysis, a system designer could determine the influence of the different parameters. An analysis of the application at hand should reveal which demands both the sensors and the sensor network as a whole have to meet. Combining these two analyses should then yield good and attainable parameter choices.

Acknowledgements We would like to thank Bert Bos (Chess) for providing us with valuable background information on sensor networks, especially concerning technical restrictions and current algorithms. We thank Malwina Luczak for her contribution to the discussions during the week of the Study group.

6.6 Bibliography

[1] Chess. Presentation, Bert Bos, "Gossiping to optimality". In *Mathematics with Industry*. TU/e, Eindhoven, The Netherlands, 30-01-2006.

[2] W. Feller. *An introduction to probability theory and its applications*. Wiley, New York, second edition, 1960.

[3] P. Fraigniaud and E. Lazard. Methods and problems of communication in usual networks. *Discrete Applied Mathematics*, 53(1-3):79–133, 1994.

[4] G. Grimmett. *Percolation*. Springer-Verlag, Berlin, second edition, 1999.

[5] S. Hedetniemi, T. Hedetniemi, and A. Liestman. A survey of gossiping and broadcasting in communication networks. *Networks*, 18:319–349, 1988.

[6] M. Ilyas and I. Mahgoub. *Handbook of Sensor Networks: Compact Wireless and Wired Sensing Systems*. CRC Press, Boca Raton, 2004.

[7] K. Pahlavan and A. Levesque. *Wireless information networks*. Wiley-Interscience, New York, NY, USA, 1995.

[8] R. T. Smythe and J. C. Wierman. *First-passage percolation on the square lattice*, volume 671 of *Lecture Notes in Mathematics*. Springer, Berlin, 1978.

[9] W. Su, E. Cayirci, and O. Akan. Overview of communication protocols for sensor networks. In *Handbook of Sensor Networks: Compact Wireless and Wired Sensing Systems*. CRC Press, Boca Raton, 2004.