International Journal on Applications of Graph Theory in Wireless Ad hoc Networks and Sensor Networks (GRAPH-HOC) Vol.9, No.1, March 2017

The Neighborhood Broadcast Problem in Wireless Ad Hoc Sensor Networks

Stefan Hoffmann and Egon Wanke

Heinrich-Heine-University Düsseldorf, Germany

Abstract

This paper considers the following Neighborhood Broadcast *problem: Distribute a message to all neighbors of a network node v under the assumption that v does not participate due to being corrupted or damaged. We present practical network protocol that can be used completely reactive. It is parameterized with a positive integer $k \in \mathbb{N}$ and it is proven to guarantee delivery for $k \geq 2d-1$, if node v is d-locally connected, which means that the set of nodes with distance between 1 and d to v induces a connected subgraph of the communication graph. It is also shown that the number of participating nodes is optimal under the restriction to 1-hop neighborhood information. The protocol is also analyzed in simulations that demonstrate very high success rates for very low values of k.*

Keywords

ad-hoc sensor network, network protocols, fault tolerance, failure recovery, neighborhood broadcast

1. Introduction

A *wireless sensor network (WSN)* consists of a large number of small devices that are deployed across a geographic area to monitor certain aspects of the environment. These sensor nodes are able to communicate with each other through a wireless communication channel. As a result of their size the resources of the sensor nodes are strongly limited in terms of available energy, which leads to a very limited range of the radio transmitters. Therefore the nodes also act as routers to relay messages on multi-hop routing paths, which have to be discovered and maintained using a routing protocol such as the ones proposed in [1, 2, 3, 4, 5, 6, 7, 8, 9].

WSNs are usually modeled as undirected communication graphs that contain one vertex for each sensor node and an edge between two vertices if and only if the corresponding sensor nodes are able to communicate with each other. From a technical point of view, modeling a WSN as a directed graph is closer to reality, because physical radio links are not necessarily symmetric [10, 11]. However, the usage of undirected graphs is well justified by the observation that low-level communication protocols for wireless transmissions currently in use, such as IEEE 802.11 or Bluetooth, require symmetry for reliable data transmission via acknowledgements.

A task that arises naturally in WSNs is the following Neighborhood Broadcast: For a sensor node v, a neighbor s of v has to transmit a message to all other neighbors of v under the assumption that v itself does not obey any of the implemented network protocols. This problem occurs during collaborative fault detection [12] where the neighbors of a sensor node v want to exchange their observations in order to decide whether v should

DOI : 10.5121/jgraphoc.2017.9101

be excluded from the network due to faults or misbehavior. The NEIGHBORHOOD BROADCAST problem also occurs during the exclusion of a misbehaving or faulty sensor node v from the network, because every neighbor of v has to be instructed not to communicate with v anymore and it can be applied, for example, to the detection algorithm presented in [13]. A practical NEIGHBORHOOD BROADCAST algorithm also has applications in reactively repairing unicast routing paths or multicast routing structures after node failures: If there is a sufficient amount of redundancy within the routing information to determine the successor(s) of a failed node v, the routing task can be accomplished by broadcasting the message across the neighborhood of v.

In this paper we present the *k-Hop Bouncing Flood (k-HBF)* network protocol, which is based on a parameter $k \in \mathbb{N}$, to distribute messages across the neighborhood of a sensor node. The k-HBF protocol can be used in a completely reactive manner, i.e. it does not need any initialization or maintenance. It only requires the nodes to know their own neighborhood. Alternatively, an optional initialization phase can be used to determine the optimal parameter k that guarantees successful delivery. Both approaches are discussed in detail in this paper. Furthermore the protocol is easy to implement in a distributed environment and therefore ready for practical applications. k-HBF is analyzed theoretically and it is proven that the protocol is guaranteed to deliver a message to every neighbor of a node v, if $k \geq 2d-1$ and v is *d-locally connected* in the communication graph G, which means that the set of nodes with hop-distance between 1 and d to v induces a connected subgraph of G. It is also shown that any invocation of the protocol results in the transmission of at most $k \cdot \Delta(G)^{k+1}$ messages, where $\Delta(G)$ is the maximum vertex degree of the communication graph. This means that, for any fixed parameter k, the k-HBF protocol only requires communication within a localized area of the network. In fact, k-HBF distributes the message to all nodes with hop-distance at most $k+1$ to v and it is also proven that this is optimal, i.e. no network protocol that is restricted to the same topology information can guarantee delivery to all neighbors of a d-locally connected node v, unless every node with hop-distance at most $2d$ to v receives at least one message.

Additionally we present a linear time algorithm to solve the LOCAL CONNECTIVITY DISTANCE (LCD)
problem, which asks for the smallest positive integer $d \in \mathbb{N}$ such that a given vertex v is d-locally connected in a given undirected graph G. This linear time algorithm is based on global topology knowledge, but it can also be implemented as a distributed algorithm in a WSN, an option that is also discussed in this paper. Therefore this algorithm can be used to determine the optimal parameter k that is required for the success of the k-HBF protocol.

Finally, an experimental analysis is performed and the simulation results demonstrate that the k-HBF protocol successfully performs a NEIGHBORHOOD BROADCAST in almost all cases for very low values of k. The computations are performed on several different network topology models commonly used in wireless network research. As a side effect, this analysis demonstrates a significant topological difference between these models.

Regarding related work: In [12] the authors focus on building data aggregation trees that span the neighborhood of a sensor node v. To discover and initially contact all neighbors of v they implicitly try to perform a NEIGHBORHOOD BROADCAST by briefly describing the usage of limited range flooding. However, sending a message with the technique they mention is only guaranteed to reach all neighbors of v, if the flooding range is high enough to flood the entire network, even in cases where it is not necessary to do so. In contrast, the k-HBF protocols uses more sophisticated extensions to the flooding protocol such as hop counter resets and allowing nodes to relay messages multiple times. The general multicast problem in wireless ad hoc networks, i.e. the task of distributing message to a known set of nodes, has received tremendous attention in the scientific community and countless proposals for protocols have been made. Therefore we refer the reader to [14, 15] for an overview on this topic. While these multicast protocols are theoretically suitable for the NEIGHBORHOOD BROADCAST task, they require the initiating node to know the neighborhood in which the message is to be distributed and this requires constant maintenance, especially when the network topology is subjected to frequent changes. And of course the entire network can be flooded to distribute a message across the neighborhood of a node, for example by using the protocols in [16, 17, 18].

The rest of the paper is organized as follows: Section 2 introduces the necessary terminology and notations before the k-HBF protocol is presented and evaluated in section 3. Section 4 is dedicated to solving the LCD problem and the experimental results are given in section 5 before the paper in concluded in section 6.

2. Basic Terminology

A pair $G = (V, E)$ is an *undirected graph* with *vertex set* V and *edge set* E, if V is a finite set and $E \subseteq \{\{u, v\} \mid u, v \in V, u \neq v\}$. The *vertex degree* $deg(v)$ of a vertex $v \in V$ is given by $deg(v) := |\{e \in E \mid v \in e\}|$ and the *maximum vertex degree of* G is denoted by $\Delta(G)$. Two distinct vertices $u, v \in V$ are called *adjacent*, if $\{u, v\} \in E$. A vertex $v \in V$ and an edge $e \in E$ are called *incident*, if $v \in e$.

A graph $G' = (V', E')$ is a *subgraph* of a graph $G = (V, E)$, if $V' \subseteq V$ and $E' \subseteq E$. For a subset of vertices $U \subseteq V$ of an undirected graph $G = (V, E)$ the graph $G|_U := (U, E|_U)$, $E|_U := \{\{u, v\} \in E \mid u, v \in U\}$ is called the *subgraph of* G *induced by* U.

A *path of length* $k-1$ *between two vertices* v_1 *and* v_k in an undirected graph is a sequence of vertices $v_1, \ldots, v_k$ such that $\forall i \in \{1, \ldots, k-1\}$: $\{v_i, v_{i+1}\} \in E$. G is *connected*, if there is a path between every pair of vertices. A *connected component* of G is a maximal induced subgraph of G that is connected. A *separation vertex* is a vertex $v \in V$ such that the induced subgraph $G|_{V \setminus \{v\}}$ has more connected components than G.

A path $u_1, \ldots, u_k$, $k \geq 3$ in an undirected graph $G = (V, E)$ is a *cycle* if $\{u_k, u_1\} \in E$. A graph without cycles is a *forest* and a connected forest is a *tree*.

The *distance* $d_G(u, v)$ between two vertices u and v in an undirected graph G is the smallest integer k such that there is a path of length k between u and v in G. The set $N_G^d(v) := \{u \in V \mid d(u, v) = d\}$ is called the *d-hop neighborhood* of a vertex $v \in V$ and $N_G^d[v] := \cup_{i=1}^k N_G^i(v)$ is the set of vertices with distance between 1 and d to v. If the graph G is obvious from the context the notations $d(u, v)$, $N^d(v)$ and $N^d[v]$ can be used instead of $d_G(u, v)$, $N_G^d(v)$ and $N_G^d[v]$.

The *diameter* of an undirected graph G is the maximum distance between any two vertices of G.

Definition 1. Let $d \in \mathbb{N}$ be a positive integer, $G = (V, E)$ an undirected graph and $v \in V$ a vertex. Vertex v is *d-locally connected* in G, if $N_G^d[v]$ induces a connected subgraph of G.
Definition 2. For an undirected graph $G = (V, E)$ and a vertex $v \in V$, the smallest positive integer $d_{min} \in \mathbb{N}$ such that v is d_{min}-locally connected in G and not $(d_{min} - 1)$-locally connected in G is called the *local connectivity distance* of v in G.

Throughout this paper the terms *node* and *vertex* are used interchangeably with *node* referring to a physical instance of a sensor node, while *vertex* refers to the mathematical counterpart within the graph model.

In section 2 a practical network protocol for the following task is presented and analyzed.

NEIGHBORHOOD BROADCAST

Given:	A WSN with undirected communication graph $G = (V, E)$ and two nodes $v \in V$ and $s \in N_G^1(v)$.
Task:	Distribute a message originating at node s to all nodes in $N_G^1(v)$ without participation of v.

During the analysis it is established that this NEIGHBORHOOD BROADCAST task is closely related to the local connectivity distance and therefore the following LCD problem is considered in section 4.

LOCAL CONNECTIVITY DISTANCE (LCD)

Given:	An undirected graph $G = (V, E)$ and a vertex $v \in V$.
Task:	Compute the local connectivity distance of v.

3. k-HOP BOUNCING FLOOD

Obviously, distributing a message M to all neighbors $N^1(v)$ of v without the participation of v itself is possible if and only if v is not a separation vertex of the communication graph $G = (V, E)$. But even if v is not a separation vertex, it might be necessary to broadcast M throughout the entire network in order to distribute it to all neighbors $N^1(v)$: Consider the cycle C_n with n vertices as a network topology:

$$C_n := (\{v_1, \ldots, v_n\}, \{\{v_i, v_{i+1} \mid 1 \leq i < n\}\} \cup \{v_n, v_1\})$$

If any one of the nodes $v_1, \ldots, v_n$ cannot be used to relay M, then the only remaining path between both neighbors of this node traverses all other nodes and therefore every multicast algorithm has to distribute M to all nodes. However, theoretical worst cases like this are rather unlikely in a real sensor network, which is why a multicast algorithm with a better performance than flooding the entire network is desirable for this scenario.

The *k-Hop Bouncing Flood (k-HBF)* protocol uses the idea of carrying a transmission counter i_t in the message header that restricts the number of retransmissions for each message: The forwarding is stopped as soon as i_t reaches the given parameter $k \in \mathbb{N}$.

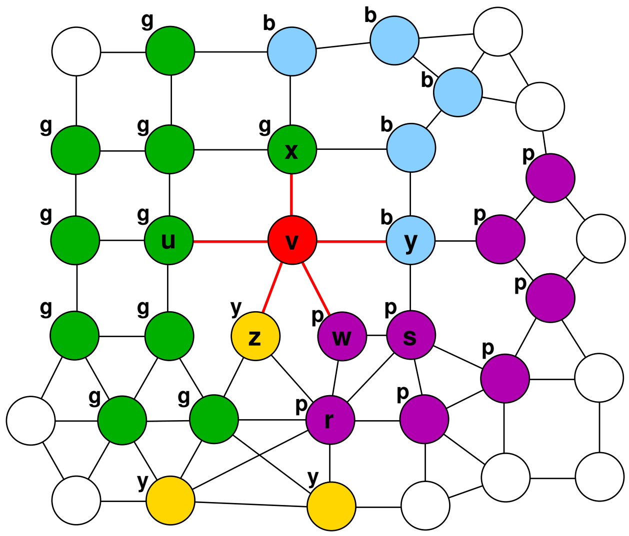

Figure 1: 2-HBF example on $G = (V, E)$: Node u initiates the protocol for its neighbor v to distribute M to $N^1_G(v) = \{u, x, y, w, z\}$. The green (g) nodes receive M due to their distance to u being at most 2 in $G|_{V\setminus\{v\}}$, the blue (b) nodes receive M due to x resetting the transmission counter i_t, the purple (p) nodes due to the reset at y and the yellow (y) nodes due to the reset at w.

This disseminates the message to all nodes of distance at most k to the node s, who initiated the protocol. This range limitation of the flooding process reduces the number of nodes transmitting the message, but the protocol is no longer guaranteed to deliver the message to all targets, as discussed above. Utilizing only information already present at the individual sensor nodes, the success probability can be increased considerably by resetting i_t to 0 at every node that is adjacent to v, which leads to M "bouncing" along the neighbors of v, see Figure 1 for an example. The idea is to distribute M not only to all nodes of distance at most k to s, but to all nodes with distance at most k to any neighbor of v or, equivalently, to every node with distance at most $k + 1$ to v. To achieve this it is not sufficient to reset i_t at every neighbor of v: For example the node r in Figure 1: Before w resets the transmission counter to 0 and sends the message to its neighbors, node r already received M with counter value 2 from s, but r is required to relay the message from w again in order to deliver it to the yellow nodes. Unlike conventional flooding it is therefore necessary that nodes can relay messages multiple times and simply marking a message M as "already seen" by a particular node to prevent infinite transmission loops by not relaying M again based on this mark is not sufficient anymore. In the k-HBF protocol, each node r that received M keeps track of the minimum $i_{min,r}$ of all transmission counter values contained in copies of M that reached r. Node r then relays M again at a later time, if and only if the new transmission counter is strictly lower than $i_{min,r}$, because in this case it might reach additional nodes. This modification guarantees that every node within the k-hop neighborhood of a node that performed a transmission counter reset will eventually receive the message, while ensuring that the transmissions terminate due to the observation that every node can transmit a message at most k times, before the minimum transmission counter for this node reaches 0.

3.1. Distributed Implementation

One of the biggest challenges for real world deployment of sensor networks is the testing and debugging of implemented algorithms in a realistic environment, which introduces additional problems that are usually not considered during simulation. In recent history there have been several real world experiments that demonstrated this issue [19, 20, 21, 22] and there has also been intensive work on testbed environments suitable to tackle this problem [23, 24, 25, 26, 27].

From this point of view, the k-HBF protocol offers the advantage of an easy and straightforward distributed implementation, which is demonstrated by the pseudo code given in Algorithm 1 for the main part of the protocol.

Algorithm 1 k-HBF protocol: Distributed implementation at node u

```
m ← 1                                              ▷ Counter for message ids
h ← HashMap                                        ▷ h : ID × NODE ↦ ℕ
procedure SendPacket(m, s, v, k, i_t, M, T)
    Pass data to link layer:
    Send (m, s, v, k, i_t, M) to all nodes in T ⊆ N^1(u)
end procedure
procedure StartHBF(k, v, M)
    m ← m + 1
    SendPacket(m, u, v, k, 0, M, N^1(u))
end procedure
procedure ReceivePacket(m, s, v, k, i_t, M)
    i_{min,u} ← ∞
    if h contains (m, s) then
        i_{min,u} ← h((m, s))
    end if
    if i_t < i_{min,u} then
        if v ∈ N^1(u) then                         ▷ neighbor resets i_t
            SendPacket(m, s, v, k, 0, M, N^1(u))
            h((m, s)) ← −∞
            return
        end if
        if i_t < k then
            SendPacket(m, s, v, k, i_t + 1, M, N^1(u))
            h((m, s)) ← i_t
        end if
    end if
end procedure
```

Packets sent by the algorithm wrap around a given message M and add some header fields required for the protocol itself in the format (message id, initiator s, target v, k, i_t, M), where the message id m is unique for each node s, meaning that (m, s) provides a unique message identifier throughout the network. The message header also contains the node v for which the protocol was initiated as well as the distance parameter k and the transmission counter i_t.

Two extensions of the protocol arise from techniques commonly used in computer networks:

To ensure long-term stability of the network, the data structure used to organize the mapping between message identifiers and minimum transmission counters at each node should be implemented based on a soft-state approach in the sense that the entries are deleted after a reasonable amount of time to avoid memory leaks.

In the next section it is shown that k-HBF is guaranteed to deliver M to all neighbors of v, if $k \geq 2d-1$ and v is d-locally connected. Therefore the optimal choice of k depends on the local connectivity distance of v, which is usually unknown. If the application is able to decide whether a broadcast attempt was successful (which is possible via acknowledgement, for example, if $|N^1(v)|$ is known or the target is one particular neighbor of v as in the case of unicast route repair), the protocol can obviously be extended by successively increasing the distance parameter k as long as the previous attempt was unsuccessful. According to common practice one would double k after each failed attempt to achieve exponential growth of the search area, which compensates for a potentially fast changing network topology.

Another possible extension that allows dynamic adaptation of the k-HBF protocol after the first execution for a node v is discussed at the end of section 4.

3.2. Analysis

This section evaluates the k-HBF protocol in terms of message complexity and success guarantee. It is shown that the number of messages transmitted by an invocation of k-HBF in a WSN with communication graph G is in $k \cdot \Delta(G)^{k+1}$. Furthermore a sufficient condition for the delivery guarantee is proven: k-HBF is guaranteed to deliver M to all neighbors of v, if $k \geq 2d-1$ and v is d-locally connected in G. Finally, it is also proven that the set of nodes participating in the k-HBF protocol is optimal with respect to all possible protocols for Neighborhood Broadcast that are restricted to the same topology information.

Theorem 1. Let $G = (V, E)$ be an undirected graph and $k \in \mathbb{N}$ a positive integer. The k-HBF protocol, initiated by a neighbor $s \in N^1(v)$ for a node $v \in V$ and a message M, transmits at most $k \cdot \Delta(G)^{k+1}$ messages.

Proof. Every neighbor $u \in N^1(v)$ performs a limited range flooding of M with range k and therefore at most $\Delta(G)^k$ nodes receive the message due to the hop counter reset at node u. Furthermore v has at most $\Delta(G)$ neighbors, which means that at most $\Delta(G)^{k+1}$ nodes receive M during the execution of the k-HBF protocol. Finally, every node w can relay M at most k times, because afterwards the minimum transmission counter $i_{min,w}$ is either 0 or $-\infty$. In conjunction it follows that $k \cdot \Delta(G)^{k+1}$ is an upper bound for the number of transmitted messages. □

The Theorem above demonstrates a theoretical worst case for number of messages transmitted the k-HBF protocol, assuming that every node in the considered part of the network actually has the maximum vertex degree $\Delta(G)$. Furthermore it assumes that the sets of nodes reached by each neighbor of v are pairwise disjoint, which is not possible, if v has more than one neighbor. Therefore the average number of transmitted messages in any real world application should be considerably lower than the presented upper bound. Of course the number of transmitted packets can be higher due to retransmissions, acknowledgements etc.

The following Lemma and the subsequent Theorem are used to establish the connection between the k-HBF protocol and the local connectivity distance of the node v it is executed for.

Lemma 1. Let $G = (V, E)$ be an undirected graph and $d \in \mathbb{N}$ a positive integer. Let $v \in V$ be a vertex that is d-locally connected in G. Then, for every pair of vertices $s, t \in N^1(v)$, there is a path $p = w_1, \ldots, w_k$ in $G|_{V\setminus\{v\}}$ between $w_1 = s$ and $w_k = t$ such that for all $i \in \{1, \ldots, k-2d+1\}$ the path $w_i, w_{i+1}, \ldots, w_{i+2d-1}$ of length $2d-1$ contains at least one vertex of $N^1(v)$.

Proof. Let $p' = u_0, \ldots, u_m$ be a shortest path between $u_0 = s$ and $u_m = t$ in $H := G|_{N^d[v]}$. Since $v \in V$ is d-locally connected such a path exists and we also know that for every vertex u on p' the distance $d_G(u, v)$ is at most d.

Path p' will now be altered successively to fulfill the required condition as follows: Let l be the minimum index such that the path $u_l, \ldots, u_{l+2d-1}$ does not contain a vertex adjacent to v. Now replace $u_l, \ldots, u_{l+2d-1}$ with a path $u_l, \ldots, s', \ldots, u_{l+2d-1}$ for a vertex $s' \in N^1(v)$ such that both paths $u_l, \ldots, s'$ and $s', \ldots, u_{l+2d-1}$ have length at most $2d-1$ as follows. This proves the Lemma, because during each iteration the length of a consecutive subsequence of the current path that violates the required condition is strictly shortened.

Let $d_l, \ldots, d_{l+2d-1}$ be the sequence D of distances to vertex v as defined by $d_i := d_G(v, u_i)$ for $i \in \{l, \ldots, l+2d-1\}$. Furthermore let $j \in \{l+1, \ldots, l+2d-1\}$ be the minimum index such that d_j is the second occurrence of this particular distance in D, meaning that there is an index $j' < j$ with $d_{j'} = d_j$. Since all distances d_i are between 2 and d by definition of p' it follows that $j \leq l + d - 1$.

Now let $v, s', q_2, \ldots, q_{d_j-1}, u_j$ be a shortest path between v and u_j in G. Note that p' does not contain vertex s', because p' contains a shortest path p'' between u_{l-1} and u_j such that p'' contains vertex $u_{j'}$. Therefore $d_H(s', u_j) = d_j - 1 < d_H(u_{l-1}, u_j)$. Then it holds that the length $L(q)$ of the path $q = u_l, \ldots, u_j, q_{d_j-1}, \ldots, q_2, s'$ is at most $2d-2$, because

$$\begin{aligned} L(q) &= L(u_l, \ldots, u_j) + L(u_j, q_{d_j-1}, \ldots, q_2, s') \\ &\leq L(u_l, \ldots, u_{l+d-1}) + d_j - 1 \leq d - 1 + d_j - 1 \leq 2d - 2. \end{aligned}$$

Furthermore it also holds that $L(q')$ for

$$q' = s', q_2, \ldots, q_{d_j-1}, u_j, u_{j+1}, \ldots u_{l+2d-1}$$

is at most $2d-1$, because

$$\begin{aligned} L(q') &= L(s', q_2, \ldots, q_{d_j-1}, u_j) + L(u_j, u_{j+1}, \ldots u_{l+2d-1}) \\ &< L(u_{l-1}, u_l, \ldots, u_j) + L(u_j, u_{j+1}, \ldots, u_{l+2d-1}) = L(u_{l-1}, u_l, \ldots, u_{l+2d-1}) = 2d. \end{aligned}$$

□

Theorem 2. Let $d \in \mathbb{N}$ be a positive integer and $v \in V$ a vertex with local connectivity distance d in an undirected graph $G = (V, E)$ that represents a WSN. Then the $(2d-1)$-HBF protocol, initiated by a neighbor $s \in N^1_G(v)$ for node v and a message M, distributes M to all nodes in $N^{2d}_G(v)$ even if v itself does not relay any messages.

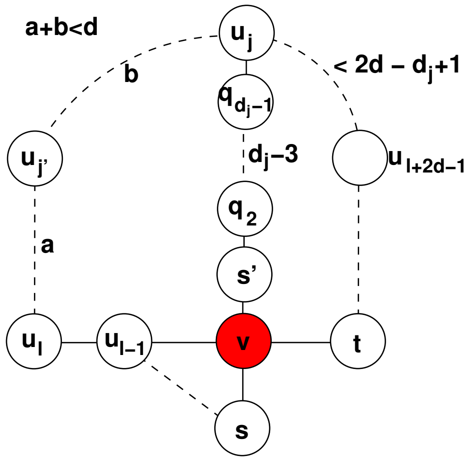

Figure 2: Proof of Lemma 1: The dashed lines represent paths of the noted lengths

Proof. Let $t \in N_G^1(v)$ be an arbitrary neighbor of v. According to Lemma 1 there is a path p between s and t in $G|_{V\setminus\{v\}}$ such that every sub-path of length $2d-1$ contains at least one neighbor of v. Therefore the message sent by s is relayed along p, because the transmission counter i_t is reset after at most $2d-1$ transmissions and hence t receives M. And since every neighbor of v receives M and resets the transmission counter once, M is distributed to all nodes with hop-distance at most $2d-1$ to any neighbor of v, i.e. to all nodes with hop-distance at most $2d$ to v. □

Although the d-hop neighborhood of v is connected, it is necessary to distribute the message across the $2d$-hop neighborhood of v, unless there is additional information about the network topology available. This is shown in the next Theorem.

Theorem 3. Let $G = (V, E)$ be an undirected graph that represents a WSN, $d \in \mathbb{N}$ a positive integer and $v \in V$ a vertex with local connectivity distance d.

Furthermore let $\mathcal{P}$ be a distributed algorithm satisfying the following properties:

1. $\mathcal{P}$ is initiated by a single node $s \in N_G^1(v)$ to distribute a message M.
2. $\mathcal{P}$ does not run on node v.
3. $\mathcal{P}$ is deterministic and the only information $\mathcal{P}$ utilizes about the network topology is the integer d and the knowledge about the 1-hop neighborhood $N_G^1(u)$ at each node u.
4. $\mathcal{P}$ terminates after a finite amount of steps and at that point every node in $N_G^1(v)$ received M.

Then $\mathcal{P}$ transmits at least one message to every node in

$$N_G^{2d}[v] \quad = \quad \bigcup_{i=1}^{2d} N_G^i(v).$$

Proof. It is first shown that every node $u \in N_G^{2d}(v)$ has to receive at least one message.

Let $H := G|_{V\setminus\{v\}}$ be the induced subgraph of G that does not contain v and consider the graph

$$G' \quad := \quad (V, E \cup \{u, v\}).$$

Vertex v is d-locally connected in G': Since

$$N_G^{2d}(v) = \bigcup_{s' \in N_G^1(v)} N_G^{2d-1}(s') = \bigcup_{s' \in N_G^1(v)} N_H^{2d-1}(s'),$$

there is a path $w_1, \ldots, w_{2d}$ between $w_1 = s'$ for some vertex $s' \in N_G^1(v)$ and $w_{2d} = u$ in H. And for all $1 \leq i \leq 2d$ it holds that $d_{G'}(v, w_i) \leq d$, because $d_H(s', w_j) \leq d - 1$ for $j \in \{1, \ldots, d\}$ and $d_H(u, w_j) \leq d - 1$ for $j \in \{d + 1, \ldots, 2d\}$.

Vertex v is not $(d-1)$-locally connected in G': Every vertex $s' \in N_G^1(v)$ satisfies $d_H(s', u) \geq 2d - 1$, because $d_G(v, u) = 2d$. Therefore every path $w_1, \ldots, w_l$ between $w_1 = s'$ for some $s' \in N_G^1(v)$ and $w_l = u$ in H satisfies $l \geq 2d$ and thus it holds that $d_H(w_d, s') \geq d - 1$ and $d_H(w_d, u) \geq d - 1$. But then there is no path between neighbor $u \in N_{G'}^1(v)$ and any neighbor $s' \in N_G^1(v)$ in $G'|_{N_{G'}^{d-1}(v)}$, meaning that v is not $(d-1)$-locally connected in G'.

We will now compare the execution of $\mathcal{P}$ initiated at node s in G to the execution of $\mathcal{P}$ initiated at node s in G': Since v is d-locally connected and not $(d-1)$-locally connected both in G and G', the integer d is identical in both executions of $\mathcal{P}$ and by property 3 these two processes can only differ from each other due to a difference in the 1-hop neighborhood information at some node. However, the 1-hop neighborhood in G and G' is identical at all nodes except for v and u and $\mathcal{P}$ does not have access to the information at node v by property 2. Therefore both executions are identical until $\mathcal{P}$ sends a message to node u in G', which happens because of property 4, and thus u also receives a message in G.

Knowing that every node in $N_G^{2d}(v)$ has to receive at least one message, it follows that every node in $N_G^{2d}[v]$ has to receive at least one message, because it is possible to generate a network topology G'' that satisfies the preconditions of the Theorem while forcing every node in $N_{G''}^{2d-1}[v]$ to relay (and therefore receive) at least one message in order to reach all nodes in $N_{G''}^{2d}(v)$: G'' contains all vertices and edges from G and for every vertex $w \in N_G^{2d-1}[v]$ with distance $x := d_G(v, w)$ one additional vertex w' that is connected to w via a path of length $2d - x$. Then $w' \in N_{G''}^{2d}(v)$ and therefore w has to receive at least one message, which implies that w and all nodes on the path between w and w' have to relay that message. □

4. The LCD Problem

After having established that the $(2d-1)$-HBF protocol successfully distributes a message within the neighborhood $N^1(v)$ of a sensor node v, if vertex v is d-locally connected in the communication graph $G = (V, E)$, this section investigates how to algorithmically determine the local connectivity distance of a vertex, i.e. the minimum integer $d > 0$, such that a given vertex v of an undirected graph is d-locally connected.

The LCD problem can obviously be solved in polynomial time by utilizing standard methods from graph theory as follows:

For a positive integer $i \in \mathbb{N}$ the neighborhood $N^i[v] = \bigcup_{j=1}^{i} N^j(v)$ can be determined in time $\mathcal{O}(|V| + |E|)$ by a slightly modified breadth first search, started at vertex v, that keeps track of the distance $d(v,u)$ for every vertex $u \in V$. Afterwards the induced subgraph $G|_{N^i[v]}$ can be constructed in time $\mathcal{O}(|V|+|E|)$ and tested for connectivity using, for example, another breadth first search. The local connectivity distance can now be computed by performing this test consecutively for all $i \in \{1,\dots,|V|\}$ in ascending order until the first connectivity test yields a positive result. The overall running time of this straightforward solution is obviously bounded by $\mathcal{O}(|V|\cdot(|V|+|E|)) = \mathcal{O}(|V|^2+|V|\cdot|E|)$.

The remainder of this section is dedicated to a more efficient solution for the LCD problem that additionally provides the possibility for a distributed implementation and therefore is more interesting in the context of WSNs. The algorithmic idea is based on the following Theorem that is formulated using the terminology introduced in the next definition.

Definition 3. Let $r \in V$ be a vertex in an undirected graph $G=(V,E)$. A *shortest path tree for G at root r* is a tree $T=(V,E_T)$, $E_T \subseteq E$ such that $d_G(r,v) = d_T(r,v)$ for all vertices $v \in V$.

A tree $T'=(V',E')$ of forest $T|_{V\setminus\{r\}}$ is called a *branch* of T. The *root of branch* T' is the (uniquely determined) vertex $v' \in N_G^1(r)$ that belongs to T', i.e. $v' \in V'$. For all vertices $v \in V\setminus\{r\}$ let $r(v)$ denote the root of the branch of T that contains v.

Furthermore, an edge $e=\{u,v\} \in E$ is called *bridge with respect to T*, if u and v belong to different branches of T

Also see Figure 3 for an example of the terms and notations introduced in this definition.

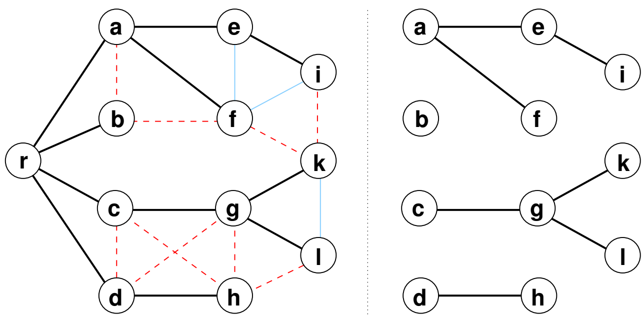

Figure 3: Example for Definition 3: **Left**: Graph $G=(V,E)$ with a shortest path tree $T=(V,E_T)$ at root r (thick black edges). The dashed red edges are bridges with respect to T. **Right**: Forest $T|_{V\setminus\{r\}}$ that defines four branches of T, induced by the vertex sets $\{a,e,f,i\}$, $\{b\}$, $\{c,g,k,l\}$ and $\{d,h\}$. The root $r(e)$ of e is vertex a, the root $r(b)$ of b is vertex b, the root $r(k)$ of k is vertex c etc.

Theorem 4. Let $G=(V,E)$ be an undirected graph, $r \in V$ a vertex and T a shortest path tree for G at root r. Furthermore let $d \in \mathbb{N}$ be a positive integer and $B \subset E$ the set of bridges with respect to T.

Every edge $e \in B$ is given a positive integer weight representing the maximum distance from r to one of the vertices incident to e. Formally, define a mapping $f: B \mapsto \mathbb{N}$ by

$$f(\{u,v\}) \quad := \quad max\{d_T(r,u), d_T(r,v)\}.$$

Additionally, define a set of edges E_1 between neighbors of r with positive integer weights

$w : E_1 \mapsto \mathbb{N}$ based on the bridges B and their weights f as follows:

$$\begin{aligned} E_1 &:= \{\{r(u), r(v)\} \mid \{u, v\} \in B\} \\ w(\{u, v\}) &:= min\{f(\{u', v'\}) \mid \{u', v'\} \in B \wedge r(u') = u \wedge r(v') = v\} \end{aligned}$$

Then vertex r is d-locally connected, if there is a subset $E' \subseteq E_1$ such that

1. the graph $(N^1_G(r), E')$ is connected and
2. $\forall e \in E' : \quad w(e) \leq d$.

Proof. Let $E' \subseteq E_1$ be a set of edges that satisfies the properties 1. and 2. above and $w, w' \in N^1_G(r)$, $w \neq w'$ two arbitrary neighbors of r. Then a path $w_1, \ldots, w_k$ in G for some positive integer $k \in \mathbb{N}$ such that $w_1 = w$, $w_k = w'$ and $\forall i \in \{1, \ldots, k\} : d_G(r, w_i) \leq d$ can be constructed as follows, meaning that r is d-locally connected in G.

Since $(N^1_G(r), E')$ is connected, there is a path between w and w' in $(N^1_G(r), E')$, i.e. there is a sequence $\{v_1, v'_1\}, \ldots, \{v_l, v'_l\}$ of edges of E' for some positive integer $l \in \mathbb{N}$ such that $v_1 = w$, $v'_l = w'$ and $\forall i \in \{1, \ldots, l-1\} : v'_i = v_{i+1}$. By definition of E_1, which is a superset of E', in conjunction with property 2. this means that there also is a sequence of bridges $\{u_1, u'_1\}, \ldots, \{u_l, u'_l\}$ satisfying the following properties:

a) $\forall i \in \{1, \ldots, l\} : \{u_i, u'_i\} \in B$

b) $\forall i \in \{1, \ldots, l-1\} : r(u'_i) = r(u_{i+1})$

c) $r(u_1) = w$

d) $r(u_l) = w'$

e) $\forall i \in \{1, \ldots, l\} : w(\{u_i, u'_i\}) \leq d$

Therefore it is sufficient to prove that, for every pair v, v' of two distinct vertices with $r(v) = r(v')$, there is a path p between v and v' in G such that every vertex u in p satisfies $d_G(r, u) \leq max\{d_G(r, v), d_G(r, v')\}$. Let p be the (unique) path between v and v' in $T|_{V\setminus\{r\}}$. Such a path exists, because v and v' belong to the same branch of T due to $r(v) = r(v')$. Additionally, every vertex u in p satisfies $d_G(r, u) \leq max\{d_G(r, v), d_G(r, v')\}$, because T is a shortest path tree for G at root r and of course p also is a path in G, because T is a subgraph of G. □

Based on Theorem the LCD problem can be solved by computing the smallest positive integer $d \in \mathbb{N}$ for which there is a set of edges $E' \subseteq E_1$ such that the conditions 1 and 2 hold. It is now shown that this can be achieved by computing a minimum spanning tree of the graph $(N^1_G(r), E_1)$ with edge weights w as defined in Theorem .

Definition 4. For an undirected graph $G = (V, E)$ with positive edge weights $w : E \mapsto \mathbb{N}$, a subgraph (V, E') of G is a *minimum spanning tree for* G, if (V, E') is connected and $\forall E'' \subseteq E$

$$(V, E'') \text{ is connected} \quad \Rightarrow \quad \sum_{e \in E''} w(e) \geq \sum_{e \in E'} w(e).$$

Minimum Spanning Tree (MST)	
Given:	An undirected graph $G = (V, E)$ with positive edge weights $w : E \mapsto \mathbb{N}$
Task:	Compute a minimum spanning tree for G.

Although the MST problem computes an edge set in which the sum of all weights is minimal while maintaining connectivity, it can also be used to compute the maximum weight necessary for achieving connectivity, because the maximum weight of an edge in a minimum spanning tree is independent of the tree itself as shown in the following Lemma.

Lemma 2. Let $G = (V, E)$ be an undirected graph with positive edge weights $w : E \mapsto \mathbb{N}$ and $T_1 = (V, E')$, $T_2 = (V, E'')$ two minimum spanning trees for G. Then $max\{w(e) \mid e \in E'\} = max\{w(e) \mid e \in E''\}$.

Proof. Let $m_1 := max\{w(e) \mid e \in E'\}$, $m_2 := max\{w(e) \mid e \in E''\}$ and let $e_1 \in E'$ be an edge with $w(e_1) = m_1$. Assume that $m_1 \neq m_2$ and, without loss of generality, let $m_1 > m_2$. Then the subgraph $(V, E' \setminus \{e_1\})$ consists of exactly two connected components induced by vertex sets V_1 and V_2. Obviously $V_1 \cup V_2 = V$ and since (V, E'') is connected there is an edge $\{v_1, v_2\} \in E''$ with $v_1 \in V_1$ and $v_2 \in V_2$. Additionally $w(\{v_1, v_2\}) \leq m_2 < m_1 = w(e_1)$ and therefore T_1 is not a minimum spanning tree, because $(V, (E' \setminus \{e_1\}) \cup \{\{v_1, v_2\}\})$ is connected and

$$\sum_{e \in E'} w(e) \quad > \quad w(\{v_1, v_2\}) - w(e_1) + \sum_{e \in E'} w(e).$$

□

According to Theorem and Lemma 2 the LCD problem can be solved using the following algorithm.

1. Compute a shortest path tree T for G at root r.
2. Determine edge set E_1 and their weights w as defined in Theorem .
3. Solve the MST problem on the graph $(N_G^1(r), E_1)$ with weights w.
4. Return the maximum edge weight that occurs in the computed minimum spanning tree.

In a centralized algorithm these steps can be implemented as follows:

1. The shortest path tree T at root r as defined in Definition 3 (where the length of a path equals the number of edges) can be computed in time $\mathcal{O}(|V| + |E|)$ by running a breadth first search on vertex r. A simple extension of this breadth first search allows the simultaneous computation of the distance $d_G(r, v)$ and the root $r(v)$ for every vertex $v \in V$, which is saved at vertex v.
2. Using the information collected in the previous step, one iteration over the edge set E is sufficient to decide whether an edge $e \in E$ is a bridge and to compute the weight $f(e)$ for all of these bridges. Let L be a list of all bridges and for a vertex $w \in N_G^1(r)$ let $p(r)$ be the position of w in the adjacency list of r. Now assign to every bridge $\{u, v\}$ in list L a key $(p(r(u)), p(r(v)))$ with $p(r(u)) < p(r(v))$, which is then used to

sort L in linear time with bucket sort. Afterwards the edge set E_1 and their weights w can be easily constructed by iterating the sorted list once more. Therefore the overall running time for this step is $\mathcal{O}(|V| + |E|)$.

3. The MST problem as defined above with integer edge weights can theoretically be solved in linear time using the trans-dichotomous minimum spanning tree algorithm presented in [28].
4. This step can obviously be done in time $\mathcal{O}(|V| + |E|)$.

The discussion above establishes the following Theorem.
Theorem 5. The LCD problem can be solved in linear time.

However, the linear time algorithm for computing integer weight minimum spanning trees in [28] is purely theoretical and not applicable in practice. If, for example, Prim's algorithm [29] is used for step 3, then the overall running of the algorithm above is in

$$\mathcal{O}(|V| + |E| + \Delta(G) \cdot log(\Delta(G))).$$

Additional to the theoretically achievable linear time implementation based on global topology knowledge, this approach is suitable for a distributed implementation in a WSN: If a node r wants to determine the minimum distance d such that the d-hop neighborhood $N_G^d[r]$ is connected, the network needs to cooperate in order to determine the edge set E_1 and the weights w. Once this is done and the collected data has been transmitted to node r, the steps 3 and 4 can be solved locally by r itself.

The shortest path tree T at root r can be built by a modified flooding algorithm, started at node r: Every transmitted message M contain a hop-counter i that is incremented after each transmission and keeps track of the distance to r. Every node v receiving such a message M updates a parent pointer to identify its parent p in the tree and notifies p that v is a child of p, if i is lower than the currently saved distance to r. Additionally M also contains the neighbor $u \in N^1(r)$ that originally transmitted the message, allowing every node v to identify the root $r(v)$ of its own branch. Based on this information it is possible to compute all bridges $e \in B$ and their weights $f(e)$ after T has been built by having every node v exchange the collected distance and branch information with all neighbors in $N^1(v)$.

Afterwards T can be used as a data aggregation tree to transmit the collected information to r while simultaneously computing the minimum weight $w(e)$ for every edge $e \in E_1$: Starting at the leafs of T the nodes send a list of edges in E_1 that result from their incident bridges to their respective parent in T. Every node that is not a leaf in T waits until it received these lists from its children in T, merges all lists including its own one by computing the minimum weight for every edge that is contained in one of the lists and sends the resulting list to its parent.

To avoid flooding the entire network, it is possible to use limited range flooding with increasing ranges to successively build larger subgraphs of T until r discovers connectivity within the considered neighborhood.

While the approach above is very efficient in terms of transmitted messages, it has to be performed proactively, because in order to determine the local connectivity distance of a node v the node itself has to participate in the computation. If this is not viable, it is also possible to determine the local connectivity distance of a node v without its participation by using the k-HBF protocol, assuming that a neighbor $s \in N^1(v)$ is capable of deciding whether an execution of k-HBF was successful: An additional counter i_{max} in the transmitted messages can keep track of the maximum value that the counter i_t ever reached before it has been reset. If every neighbor $u \in N^1(v)$ computes the minimum value $k_{min}(u)$ of the i_{max} counters of all messages u received and transmits the result back to the initiating node s, then s is capable of determining the minimum value k_{min} (and therefore the local connectivity distance of v) such that the k_{min}-HBF protocol succeeds via $k_{min} = max\{k_{min}(u) \mid u \in N^1(v) \setminus \{s\}\}$. By distributing the value k_{min} to all nodes in $N^1(v)$, the protocol can be extended to provide increased performance during future executions for the same node v.

5. Experimental Analysis

Based on the theoretical analysis in section , the experimental analysis of the k-HBF protocol is performed by solving the LCD problem rather than simulating the protocol itself. The computations are done on randomly generated graph using different graph models that are commonly used for simulations in the wireless network research community. The simulation software has been implemented in Java and executed in the Java Runtime Environment at version 8u74 on a system running the Kubuntu 12.04 LTS operating system, whose NativePRNG was used for random number generation.

For every set of parameters the results are computed for 100 graphs that are generated as follows:

1. 9000 vertices are placed randomly in a 3000×3000 square.
2. Edges are added according to one of the graph models described below.
3. The connected component containing the maximum number of vertices is determined and used for the simulation.

To generate edges the following models are used:

1. Unit Disk Graph (UDG) with radius $r \in \mathbb{R}$: An edge between two vertices is generated if and only if their euclidean is at most r. It is well known that this graph model is unrealistic for real life wireless networks and yet it is still used by many researchers for simulations. We use this model as a point of reference.
2. Waxman with parameters $\alpha \in [0, 1]$ and $r \in \mathbb{R}$: The random graph model introduced by Waxman in [30] captures important effects that occur in real life networks. Unlike the UDG model it does not guarantee the existence of edges between vertices that are positioned close to each other and it also generates “long” edges. The Waxman model is often used by researchers, because it is implemented in the BRITE topology generator that can easily be used in conjunction with the *ns-3* network simulator. In this model an edge between two vertices u, v is added to the graph with probability
$$P(u, v) \quad = \quad \alpha \cdot exp\left(\frac{-d(u, v)}{0.5 \cdot \sqrt{2} \cdot r}\right)$$
where $d(u, v)$ denotes the euclidean distance between u and v. The scaling factor of $0.5 \cdot \sqrt{2}$ has been chosen such that, for $\alpha = 1$ and the values used for r, the average

vertex degree of the generated graphs is similar to the UDG model with the same parameter r.

3. Locality with parameter $r \in \mathbb{R}$: The locality model that is mentioned in [31] and also implemented in the BRITE topology generator partitions the euclidean distances between two vertices into a finite amount of categories and assigns different, constant probabilities to each category. Based on the parameter r and the euclidean distance $d(u, v)$ between u and v, the edge $\{u, v\}$ is added with probability

$$P(u, v) = 0.8 - 0.1 \cdot (i - 1),$$

if $r \cdot \frac{i-1}{4} < d(u, v) \leq r \cdot \frac{i}{4}$ for $i \in \{1, \dots, 8\}$. Edges between vertex pairs u, v with $d(u, v) > 2r$ are not added. Unlike the UDG model, edges between vertices that are positioned close to each other are not guaranteed while there still is an upper bound for the distance between adjacent vertices and in contrast to the Waxman model the probability does not decrease exponentially with the distance.

For every set of parameters, the *average vertex degree* δ and *the average graph diameter* $\varnothing$ are determined for comparability. Then, for every distance d_{min}, it is computed how many of the n *vertices* are d_{min}-locally connected, but not $(d_{min} - 1)$-locally connected, i.e. have local connectivity distance d_{min}.

Two special cases are noted separately in the following tables: The *number of separation vertices* is given in row $d_{min} = \infty$, because these vertices are obviously never d-locally connected and it is not possible to perform a Neighborhood Broadcast for them. Also the *number* Δ_1 *of vertices with vertex degree* 1 is given separately due to the fact that they are trivially 1-locally connected. The Δ_1 vertices are also contained in the number given for $d_{min} = 1$.

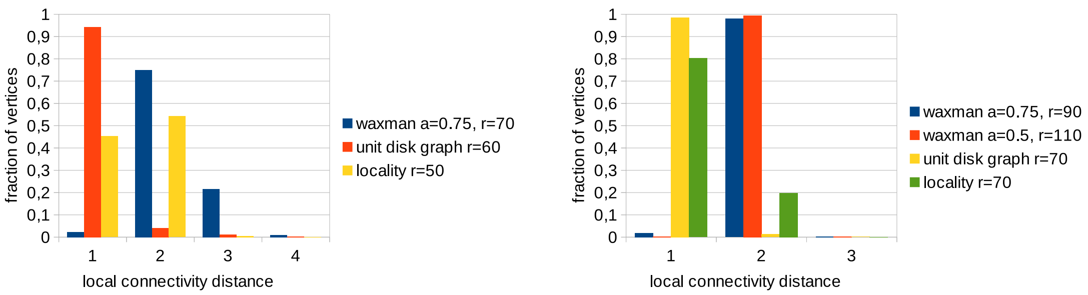

Figure 4: Distribution of local connectivity distances in graphs with similar vertex degree: **Left:** Waxman: $\delta = 11.06$, UDG: $\delta = 11.12$, Locality: $\delta = 9.79$ **Right:** Waxman $\alpha = 0.75$: $\delta = 16.98$, Waxman $\alpha = 0.5$: $\delta = 17.78$, UDG: $\delta = 15.10$, Locality: $\delta = 19$

Since the existence of edges between vertices that are positioned close to each other are guaranteed and the fact that the maximum length of any edge is bounded in the purely theoretical unit disk graph model, the Tables 5 and 6 exhibit that the majority of vertices in this model is 1-locally connected, even in very sparse graphs. All graphs generated based on the Waxman model contain a significantly lower rate of 1-locally connected vertices and the number of vertices clearly spikes at local connectivity distance 2. In the locality model on the other hand, the spike gradually shifts from local connectivity distance 2 to local

Table 1: Waxman Model

α	1
r	50
n	899170
δ	7.63
$\varnothing$	37.38
Δ_1	4556
d_{min}	#vertices (%)
1	34228 (3.80)
2	797634 (88.7)
3	61867 (6.88)
$4-6$	781 (0.09)
∞	4660 (0.52)

α	1
r	70
n	899987
δ	14.75
$\varnothing$	22.07
Δ_1	70
d_{min}	#vertices (%)
1	41529 (4.61)
2	854816 (94.98)
3	3569 (0.40)
4	2 (0.00)
∞	71 (0.01)

α	1
r	90
n	900000
δ	24.09
$\varnothing$	15.8
Δ_1	1
d_{min}	#vertices (%)
1	60066 (6.67)
2	839379 (93.26)
3	554 (0.06)
∞	1 (0.00)

Table 2: Waxman Model

α	1
r	110
n	900000
δ	35.54
$\varnothing$	12.14
Δ_1	0
d_{min}	#vertices (%)
1	81543 (9.10)
2	818301 (90.92)
3	156 (0.02)
∞	0 (0.00)

α	0.75
r	50
n	895888
δ	5.74
$\varnothing$	42.2
Δ_1	18821
d_{min}	#vertices (%)
1	20015 (2.23)
2	657711 (73.41)
3	189849 (21.19)
4	8373 (0.93)
5	331 (0.04)
6	28 (0.00)
∞	19581 (2.19)

α	0.75
r	70
n	899928
δ	11.06
$\varnothing$	24.18
Δ_1	578
d_{min}	#vertices (%)
1	12104 (1.34)
2	872618 (96.97)
3	14594 (1.62)
4	30 (0.00)
∞	582 (0.06)

Table 3: Waxman Model

α	0.75
r	90
n	899997
δ	18.07
$\varnothing$	16.98
Δ_1	30
d_{min}	#vertices (%)
1	16325 (1.81)
2	881550 (97.95)
3	2092 (0.23)
∞	30 (0.00)

α	0.75
r	110
n	900000
δ	26.66
$\varnothing$	13.06
Δ_1	0
d_{min}	#vertices (%)
1	25332 (2.81)
2	874132 (97.13)
3	536 (0.06)
∞	0 (0.00)

α	0.5
r	50
n	874040
δ	3.92
$\varnothing$	51.83
Δ_1	74781
d_{min}	#vertices (%)
1	16385 (1.87)
2	352649 (40.35)
3	328613 (37.60)
4	80075 (9.16)
$5-9$	15109 (1.73)
∞	81209 (9.29)

Table 4: Waxman Model

α	0.5
r	70
n	898911
δ	7.38
$\varnothing$	28.04
Δ_1	5940
d_{min}	#vertices (%)
1	4847 (0.54)
2	770761 (85.74)
3	116337 (12.94)
4	906 (0.10)
5	23 (0.00)
∞	6037 (0.67)

α	0.5
r	90
n	899944
δ	12.03
$\varnothing$	19.1
Δ_1	377
d_{min}	#vertices (%)
1	2012 (0.22)
2	883397 (98.16)
3	14144 (1.57)
4	8 (0.00)
∞	383 (0.04)

α	0.5
r	110
n	899996
δ	17.78
$\varnothing$	14.56
Δ_1	31
d_{min}	#vertices (%)
1	2103 (0.23)
2	894926 (99.44)
3	2934 (0.33)
∞	33 (0.00)

Table 5: Unit Disk Graph Model

r	40
n	784287
δ	5.13
$\varnothing$	222.63
Δ_1	20140
d_{min}	#vertices (%)
1	531542 (67.78)
2	71115 (9.07)
$3-6$	67179 (8.57)
$7-20$	36789 (4.69)
$21-50$	9104 (1.16)
$51-80$	1568 (0.20)
$81-121$	389 (0.05)
∞	66601 (8.49)

r	50
n	897819
δ	7.76
$\varnothing$	118.13
Δ_1	3307
d_{min}	#vertices (%)
1	750717 (83.62)
2	67635 (7.53)
3	32973 (3.67)
4	17447 (1.94)
5	9494 (1.06)
$6-10$	12118 (1.35)
$11-21$	840 (0.09)
∞	6595 (0.73)

r	60
n	899868
δ	11.12
$\varnothing$	89.56
Δ_1	320
d_{min}	#vertices (%)
1	847605 (94.19)
2	36412 (4.05)
3	10895 (1.21)
4	3153 (0.35)
$5-11$	1254 (0.14)
∞	549 (0.06)

Table 6: Unit Disk Graph Model (left) / Locality Model (right)

r	70
n	899983
δ	15.10
$\varnothing$	73.37
Δ_1	39
d_{min}	#vertices (%)
1	886854 (98.54)
2	11383 (1.26)
3	1470 (0.16)
∞	62 (0.01)

r	90
n	900000
δ	31.16
$\varnothing$	28.91
Δ_1	0
d_{min}	#vertices (%)
1	861893 (95.77)
2	38107 (4.23)
∞	0 (0.00)

Table 7: Locality Model

r	30		r	50		r	70	
n	790951		n	899844		n	899996	
δ	3.75		δ	9.79		δ	19.00	
$\varnothing$	190.64		$\varnothing$	61.59		$\varnothing$	39.14	
Δ_1	69865		Δ_1	933		Δ_1	10	
d_{min}	#vertices	(%)	d_{min}	#vertices	(%)	d_{min}	#vertices	(%)
1	171022	(21.62)	1	407176	(45.25)	1	722197	(80.24)
2	365517	(46.21)	2	487215	(54.14)	2	177787	(19.75)
3	56484	(7.14)	3	4190	(0.47)	3	2	(0.00)
$4-10$	65442	(8.27)	4	268	(0.03)	∞	10	(0.00)
$11-40$	20853	(2.64)	5	12	(0.00)			
$41-102$	1861	(0.23)	6	7	(0.00)			
∞	109772	(13.88)	∞	976	(0.11)			

connectivity distance 1 with increasing vertex degree, presumably due to the still existing maximum edge length in this model.

Measuring the success rate of the k-HBF protocol as a percentage of non-separation vertices, the conducted simulations indicate a success rate of more than 80% across all considered graphs models for the 5-HBF protocol, the minimum being defined by the very sparse graphs with an average vertex degree below 6. Restricted to graphs with an average vertex degree of at least 7, the success rate of the 5-HBF protocol is above 95% and the success rate of the 3-HBF protocol is still above 85%.

While one might intuitively presume that most of the vertices with higher local connectivity distance are close to the border of the geometric region that graph is placed across, the sample graphs taken during the simulation do not verify this conjecture: Typically these vertices offer some sort of "shortcut" through an otherwise sparse region of the graph, see Figure 5 for an example.

6. Conclusion

A protocol that offers the possibility of performing a Neighborhood Broadcast has applications in several areas of ad hoc wireless networking such as fault tolerance, routing and security. The presented k-HBF protocol is easy to implement and has been proven to distribute a message successfully across the neighborhood of a non-cooperating node v, if the parameter k is chosen to be at least $2d-1$ with d being the local connectivity distance of v and it has been shown that the set of participating nodes is optimal.

The local connectivity distance of a vertex v can be computed in linear time based on global topology knowledge or proactively in an ad hoc wireless network with a reasonable amount of transmitted messages. This knowledge can be used to optimize future executions of the k-HBF protocol or as a metric for redundancy and the importance of v within the network. Alternatively the local connectivity distance can also be determined reactively by the k-HBF protocol itself, which provides the possibility of a dynamically adapting protocol.

The presented simulations demonstrate that the k-HBF protocol can be expected to pro-

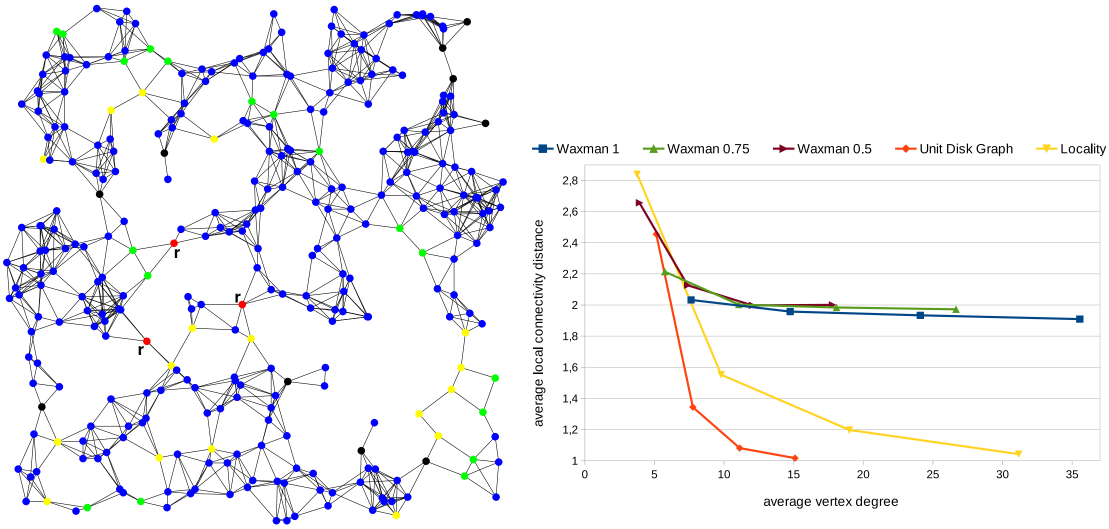

Figure 5: **Left:** An induced subgraph of a unit disk graph with radius $r = 50$: The red (r) vertices have local connectivity distance 5 due to their position between two "holes". Other distances are blue ($d_{min} = 1$), green ($d_{min} = 2$), yellow ($d_{min} = 3$) and black ($d_{min} > 5$). Note that the local connectivity distance of some vertices at the border is not verifiable based on this image, because some incident edges have been removed. **Right:** Topological difference between graph models: While the average local connectivity distance quickly approaches 1 with increasing vertex degree for the idealized unit disk graph and locality models, the more realistic Waxman model exhibits an average local connectivity distance of about 2 for reasonably dense networks.

vide very high success rates in real world networks for values of k as low as 3 to 5. As a side effect, the results clearly show a significant, topological difference between graphs generated by the idealized unit disk graph model and more realistic network models, further strengthening the known recommendations that unit disk graphs should not be used to model wireless networks.

7. References

[1] A. Awad, S. Lei, R. German, and F. Dressler. Advantages of virtual addressing for efficient and failure tolerant routing in sensor networks. In *Wireless On-Demand Network Systems and Services, 2009. WONS 2009. Sixth International Conference on*, pages 111–118, 2009.

[2] P. Bose, P. Morin, I. Stojmenovic, and J. Urrutia. Routing with guaranteed delivery in ad hoc wireless networks. *Wireless Networks*, 7(6):609–616, 2001.

[3] A. Caruso, S. Chessa, S. De, and A. Urpi. Gps free coordinate assignment and routing in wireless sensor networks. In *INFOCOM* [32], pages 150–160.

[4] E. H. Elhafsi, N. Mitton, and D. Simplot-Ryl. Cost over progress based energy efficient routing over virtual coordinates in wireless sensor networks. In *WOWMOM*, pages 1–6. IEEE, 2007.

[5] Q. Fang, J. Gao, L.J. Guibas, V. de Silva, and L. Zhang. GLIDER: gradient landmark-based distributed routing for sensor networks. In *INFOCOM* [32], pages 339–350.

[6] R. Fonseca, S. Ratnasamy, J. Zhao, C.T. Ee, D.E. Culler, S. Shenker, and I. Stoica. Beacon vector routing: Scalable point-to-point routing in wireless sensornets. In *NSDI*. USENIX, 2005.

[7] D. Gaußmann, S. Hoffmann, and E. Wanke. Hierarchical bipartition routing for delivery guarantee

in sparse wireless ad hoc sensor networks with obstacles. In *ICWN*, pages 3–9. CSREA Press, 2012.

[8] J. Li, L. Gewali, H. Selvaraj, and M. Venkatesan. Hybrid greedy/face routing for ad-hoc sensor network. In *DSD*, pages 574–578. IEEE, 2004.

[9] M.-J. Tsai, H.-Y. Yang, B.-H. Liu, and W.-Q. Huang. Virtual-coordinate-based delivery-guaranteed routing protocol in wireless sensor networks. *IEEE Transactions on Networking*, 17(4):1228–1241, 2009.

[10] A. Cerpa, J. Wong, L. Kuang, M. Potkonjak, and D. Estrin. Statistical model of lossy links in wireless sensor networks. In *IPSN*, pages 81–88. IEEE, 2005.

[11] D. Kotz, C. Newport, and C. Elliott. The mistaken axioms of wireless-network research. Technical report, Dept. of Computer Science, Dartmouth College, July 2003.

[12] G. Wang, W. Zhang, Guohong Coa, and T. La Porta. On supporting distributed collaboration in sensor networks. In *Military Communications Conference, 2003. MILCOM '03. 2003 IEEE*, volume 2, pages 752–757 Vol.2, Oct 2003.

[13] J. Clulow and T. Moore. Suicide for the common good: A new strategy for credential revocation in self-organizing systems. *SIGOPS Oper. Syst. Rev.*, 40(3):18–21, July 2006.

[14] E.M. Royer andC.K. Toh. A review of current routing protocols for ad hoc mobile wireless networks. *Personal Communications, IEEE*, 6(2):46–55, Apr 1999.

[15] L. Junhai, Y. Danxia, L. Xue, and F. Mingyu. A survey of multicast routing protocols for mobile ad-hoc networks. *Communications Surveys Tutorials, IEEE*, 11(1):78–91, First 2009.

[16] A. Qayyum L. and Viennot and A. Laouiti. Multipoint relaying for flooding broadcast messages in mobile wireless networks. In *System Sciences, 2002. HICSS. Proceedings of the 35th Annual Hawaii International Conference on*, pages 3866–3875, 2002.

[17] Y. Sasson, D. Cavin, and A. Schiper. Probabilistic broadcast for flooding in wireless mobile ad hoc networks. In *2003 IEEE Wireless Communications and Networking, WCNC 2003, New Orleans, LA, USA, 16-20 March, 2003*, pages 1124–1130. IEEE, 2003.

[18] M. Sheng, J. Li, and Y. Shi. Relative degree adaptive flooding broadcast algorithm for ad hoc networks. *TBC*, 51(2):216–222, 2005.

[19] K. Chin, J. Judge, A. Williams, and R. Kermode. Implementation experience with MANET routing protocols. *Computer Communication Review*, 32(5):49–59, 2002.

[20] S. Desilva and S. Das. Experimental evaluation of a wireless ad hoc network. In Ton Engbersen and E. K. Park, editors, *Proceedings Ninth International Conference on Computer Communications and Networks, ICCCN 2000, 16-18 October 2000, Las Vegas, Nevada, USA*, pages 528–534. IEEE, 2000.

[21] K. Langendoen, A. Baggio, and O. Visser. Murphy loves potatoes: experiences from a pilot sensor network deployment in precision agriculture. In *20th International Parallel and Distributed Processing Symposium (IPDPS 2006), Proceedings, 25-29 April 2006, Rhodes Island, Greece*. IEEE, 2006.

[22] E. Nordström, P. Gunningberg, and H. Lundgren. A testbed and methodology for experimental evaluation of wireless mobile ad hoc networks. In Roberto Battiti, Mario Gerla, and Piero Spinnato, editors, *1st International Conference on Testbeds & Research Infrastructures for the DEvelopment of NeTworks & COMmunities (TRIDENTCOM 2005), 23-25 February 2005, Trento, Italy*, pages 100–109. IEEE Computer Society, 2005.

[23] S. Jadhav, T. Brown, S. Doshi, D. Henkel, and R. George. Lessons learned constructing a wireless ad hoc network test bed. In *1st International Workshop on Wireless Network Measurements*, 2005.

[24] Z. Jin, B. Liang, Y. Shu, and O. Yang. Designing and implementing a mobile ad hoc network testbed. In *Electrical and Computer Engineering, 2002. IEEE CCECE 2002. Canadian Conference on*, volume 3, pages 1559–1564 vol.3, 2002.

[25] H. Lundgren, D. Lundberg, J. Nielsen, E. Nordström, and C. Tschudin. A large-scale testbed for reproducible ad hoc protocol evaluations. In *2002 IEEE Wireless Communications and Networking Conference Record, WCNC 2002, Orlando, Florida, USA, MArch 17-21, 2002*, pages 412–418. IEEE, 2002.

[26] D. Maltz, J. Broch, and D. Johnson. Experiences designing and building a multi-hop wireless ad hoc network testbed. Technical Report CMU-CS-99-116, CMU School of Computer Science, March 1999.

[27] Y. Zhang and W. Li. An integrated environment for testing mobile ad-hoc networks. In *Proceedings of the 3rd ACM Interational Symposium on Mobile Ad Hoc Networking and Computing, MobiHoc 2002, June 9-11, 2002, Lausanne, Switzerland*, pages 104–111. ACM, 2002.

[28] M. Fredman and D. Willard. Trans-dichotomous algorithms for minimum spanning trees and shortest paths. *Journal of Computer and System Sciences*, 48(3):533 – 551, 1994.

[29] R. Prim. Shortest connection networks and some generalizations. *Bell System Technical Journal, The*, 36(6):1389–1401, Nov 1957.

[30] B.M. Waxman. Routing of multipoint connections. *Selected Areas in Communications, IEEE Journal on*, 6(9):1617–1622, Dec 1988.

[31] E.W. Zegura, K.L. Calvert, and S. Bhattacharjee. How to model an internetwork. In *INFOCOM '96. Fifteenth Annual Joint Conference of the IEEE Computer Societies. Networking the Next Generation. Proceedings IEEE*, volume 2, pages 594–602 vol.2, Mar 1996.

[32] *INFOCOM 2005. 24th Annual Joint Conference of the IEEE Computer and Communications Societies, 13-17 March 2005, Miami, FL, USA*. IEEE, 2005.