

Interference Aware Construction of Multi- and Convergecast Trees in Wireless Sensor Networks

Tomas Johansson, Evgeny Osipov, and Lenka Carr-Motyčková

Department of Computer Science and Electrical Engineering,
Luleå University of Technology
{Tomas.Johansson,Evgeny.Osipov,Lenka.Carr}@ltu.se

Abstract. In this paper we consider a problem of building a forwarding tree for multicast and convergecast traffic in short-range wireless sensor networks. Interference awareness and energy efficiency are the major design objectives for WSN protocols in order to maximize the network lifetime. The existing multicast algorithms aim at constructing low-energy cost trees. Adding interference-awareness, however, leads to increased throughput and further reduces the energy consumption by avoiding unnecessary retransmissions due to interference-induced packet losses. We propose a Localized Area-Spanning Tree (LAST) protocol for wireless short-range sensor networks. Unlike previous similar protocols, the LAST protocol reaches all the nodes in a given geographical area, rather than only specific individual nodes. When creating the tree, the protocol jointly optimizes the energy cost and the interference imposed by the structure.

Keywords: multicast, convergecast, wireless sensor networks, interference.

1 Introduction

Over the recent years wireless sensor networks (WSN) appeared as a unique networking environment with respect to routing amongst other aspects. In this paper we focus on constructing geographic based multicast and convergecast trees. In the context of wireless sensor networks, the converge cast traffic patterns are particularly important where data from a set of nodes at different locations should be transmitted to a single processing/control unit. The multicast communication is essential when the control unit sends a binary file with updated functionality to a group of nodes.

While multicast communications is a well researched area with multiple available solutions in the computer-oriented Internet, there are many challenging and yet unsolved problems in wireless sensor networks. Firstly, in comparison to address-centric computer networks it is difficult, and in many applications impossible, to assign hierarchically structured addresses to sensor nodes. Therefore, attribute-based and geographic based addressing is most common in WSNs. The second set of problems arises from specifics of wireless communications. The

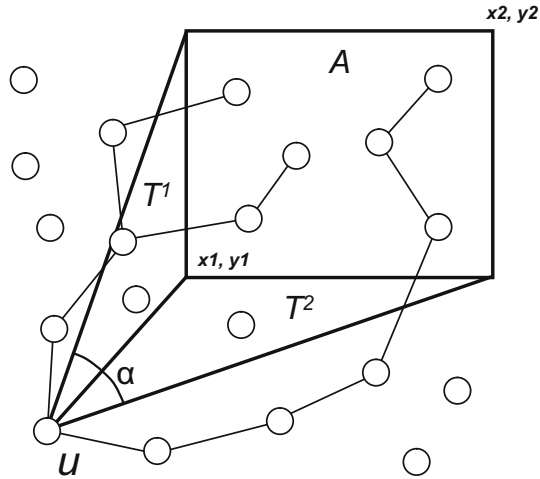


Fig. 1. The objective of LAST: Create an energy efficient multicast tree towards a geographic region. Energy cost is estimated as proportional to the area of the target area A in addition to the area of the two triangles T^1 and T^2 .

major problem here is the broadcast radio transmission medium that introduces a phenomenon of cross-node interference when neighboring nodes transmit simultaneously. Most significantly, radio interference manifests itself in reduction of data flows throughput. In the context of energy constrained wireless sensor nodes, the interference dramatically reduces the network's lifetime¹ as well. The ultimate design objective for all WSN protocols, and multi- and convergecast in particular, is to minimize energy consumption in the network. This can be done either directly by aiming at keeping the communications related energy cost as low as possible, or indirectly by reducing the cross-link interference through topology control for minimizing the number of unnecessary retransmissions.

In this paper we introduce LAST, a Localized Area-Spanning Tree protocol. The main application of the algorithm is to request and collect data from all sensor nodes located in a specified area, as shown in Figure 1. When building the tree, the LAST protocol jointly optimizes the energy consumption and the induced interference of the tree. Overall, the protocol achieves better energy conservation in comparison to existing approaches. In short our protocol works as follows: each node is aware of its geographic position and the positions of its immediate neighbors. All nodes are also able to compute a distance based interference metric. When building the tree each node is given a task to cover a subregion of the target region by choosing one or several immediate neighbors that minimize: (a) a geographical distance towards the subregion; and (b) the energy consumption while going through a particular neighbor according to the interference metric.

¹ The lifetime of a network is defined as the time until the first node in the network runs out of energy.

Even though one of the objectives is to minimize the interference, creating a tree between the control unit and the target area rather than just a path would intuitively seem to increase the interference since the different branches would interfere with each other. However, the LAST algorithm takes both interference and energy cost into account when deciding whether or not to branch out at any given node. We define the angle α in Figure 1 to be u 's *angle of visibility* of the target area A. If the angle of visibility is small, the algorithm is unlikely to introduce branches in the structure. On the other hand, if the angle is large, introducing branches would mean that the energy consumption of the tree would be reduced, while the branches would be largely separate from each other.

The remainder of the paper is structured as follows. Section 2 overviews the related work in the area of multicast topologies construction algorithms. Section 3 present the network model and our assumptions, and in Section 4 we describe the LAST algorithm, including how interference and energy cost are measured and estimated. Section 5 presents a discussion as well as directions for our future work. We conclude the paper in Section 6.

2 Related Work

Multicast protocols can use both grid-based and tree-based structures to distribute the data from a sender to the receivers. In a tree-based structure, there is only one unique path between any sender-destination pair, while there may be several in a grid-based structure.

One example of an algorithm that creates a tree structure is GMP, [9] a distributed Geographic Multicast Protocol, where the protocol finds paths to destination nodes given their geographical coordinates. This is done by calculating virtual euclidian Steiner trees, which the actual paths are based on. The aim of the algorithm is to minimize the number of hops as well as the energy consumption.

GMR [7], the Geographic Multicast Routing protocol, is another example of tree-based multicast. The objective of this algorithm is to create a tree that reaches a set of nodes at certain given geographical coordinates. This is done by introducing the *cost over goodness* ratio, where the cost is the number of neighbors a node uses to forward the message, and goodness is the reduction of the total remaining distance to the destinations. The advantages of this algorithm is a low computation time as well as a lower number of transmissions than comparable algorithms.

In [5], the GMR protocol is extended into the multicast protocol HGMR (Hierarchical Geographic Multicast Routing). The network area is divided into cells, where one node in each cell is the designated access point for the cell. The hierarchical structure reduces the encoding overhead, while the protocol retains the forwarding efficiency of the GMR protocol.

MSTEAM, another multicast protocol with the goal to create a tree to a number of nodes with specified coordinates is presented in [2], where the first step is to create a minimum spanning tree over the target nodes. The next step

is to create the actual multicast tree, and the decision when to branch is taken locally. Compared to similar protocols, the MSTEAM protocol performs well in simulations with regards to the energy efficiency.

Zeng et al. [10] introduce a grid multicast protocol, which unlike the previous algorithms is focused only on minimizing the energy cost, and not the hop count.

One common theme for the existing multicast protocols is that they focus on minimizing the energy consumption directly, but not indirectly through minimizing the interference of the resulting path structure.

In comparison to multicast, convergecast introduces the additional challenge of avoiding packet collisions when several sensors transmit data simultaneously. In [11], the authors introduce a block acknowledgment scheme in order to guarantee continuous packet forwarding.

3 Network Model and Assumptions

3.1 Model

A sensor network is modelled as an Euclidian graph $G = (V, E)$ with the vertices in V representing sensor nodes, and the edges in E representing communication links. The euclidian position (x, y) of the vertices in the graph corresponds to the physical position of the nodes in the euclidian two dimensional space, which means that the edge weight $w(u, v)$ represents the physical distance between nodes u and v . The energy cost to transmit a data unit from node u to its one-hop neighbor v is designated $c(u, v)$. Each node u has a maximum transmission range R_u .

In this paper, we discuss a nested structure of the target geographical region. The following notation is used: an area A can be divided into n subareas designated A_i^1 , where $1 \leq i \leq n$. The subareas do not have to be disjoint, but the union of all subareas is identical to the original area A . The superscript indicates that we are at the first level below the original area A . Subarea A_i^1 can in its turn be divided further into m subsubareas A_{ij}^2 , where $1 \leq j \leq m$. This process can be repeated any number of steps.

3.2 Assumptions

Throughout the paper, we make assumptions on the network as follows:

Bidirectional links. We assumed the sensor nodes to be equal. However, depending on the environment equal nodes in different locations will have different transmission ranges. This means that some of the links in the network will be unidirectional. In order to simplify the presentation of the algorithm, we will in this paper assume that the links in the network are bidirectional. However, the algorithm can easily be modified to handle unidirectional links as well.

Variable transmission power. We assume that all nodes can adjust their transmission power to any value from 0 to their maximum transmission power, depending on the desired transmission radius: when transmitting to node v , node u uses the lowest possible transmission power needed to reach v . A common path

loss model says that the signal strength received by a node can be described as p/d^α , where p is the transmission power used by the sending node, d is the distance between two nodes, and α is an environment-specific path loss exponent.

Connectivity. We assume that the sensor network does not contain any nodes where packets can get stuck using multi-hop greedy geographic forwarding. In other words, for any node v in the network and any location l outside v 's transmission range, v has at least one one-hop neighbor that is nearer to l than v is.

Topology knowledge. Each node knows only the positions of all its one-hop neighbors as well as the amount of signal loss when communicating with them. In reality, this information can be gathered by overhearing beacon transmissions between the neighboring nodes. We also assume that the nodes are global-topology agnostic. A detailed description of the information gathering is outside the scope of this paper.

Uniform distribution. In the beginning of the network lifetime, the nodes are assumed to be evenly distributed over the deployment area instead of clustered. This assumption holds until nodes start to run out of power.

4 Metrics and Algorithm

4.1 Metrics

Interference metrics. Unless the traffic patterns in a network is known in advance, the amount of interference can only be modelled on the properties of the network topology. Several different interference metrics have been proposed. In [1], the interference of a network is defined as the *maximum edge coverage* occurring in the network, where the *coverage* ζ of an edge $e = (u, v)$ is defined as the number of nodes that are within distance $|u, v|$ to at least one of u and v , or more formally:

$$\zeta(e) = |\{w \in V, |v, w| \leq |u, v| \cup |u, w| \leq |u, v|\}| \quad (1)$$

and

$$MaxEdgeCoverage(G) = \max_{e \in E} \zeta(e)$$

Even though (1) implicitly assumes that the transmission area is circular for all nodes, this assumption is not necessary in order to compute the coverage metric. In the remainder of this paper, the coverage of a given link (u, v) is defined as the number of nodes that are affected by the transmission over that link, regardless of their distance to u and v .

In [8] the authors claim that defining interference as the maximum edge coverage in the network is problematic, since it is sender-centric, and since small changes in the network can drastically change the interference measure. In [3], we discuss another problem with the maximum edge coverage metric: it does not take the length of the paths in the graph into account. Removing edges with high interference, in order to reduce the interference of a graph, can cause the

length of the paths between node pairs to grow indefinitely. Sending a message over a long path will then lead to a high total interference, even if each link in that path would result in low coverage. Based on these issues, we proposed to define the interference $TotI(P)$ of a path P to be the sum of the coverage of all edges in that graph.

$$TotI(P) = \sum_{e \in P} \zeta(e) \quad (2)$$

Unlike the previous metric, this metric reflects the total amount of interference when sending over a path. Removing edges with high coverage from the network is advantageous only if an alternative path with lower interference exists. The interference definition in this paper is similar to our previous definition: the total interference of a set of links is equal to the sum of the coverage of all links in the set.

Based on our assumptions on the topology knowledge, a node u can compute which neighbors will be affected when it communicates with a one-hop neighbor v . In other words, node u can compute the edge coverage of any link (u, v) , and the interference $\zeta(u, v)$ can be computed locally.

Energy cost metrics. Since a node has no global topology knowledge, the energy cost cannot be computed directly. Based on our assumption on uniform distribution of nodes in the deployment field, we estimate the energy cost of a tree spanning a region as proportional to the area of the region, as illustrated in Figure 1.

Thus, the energy cost EC used in LAST to cover an area A from a given location is a function of the area of the A together with the area of the two triangles, as seen in the figure.

$$EC \propto A_{AREA} + T_{AREA}^1 + T_{AREA}^2. \quad (3)$$

When comparing energy costs for covering different areas, the size of the areas can therefore be compared directly.

4.2 Localized Area-Spanning Tree

In this section, we present the operations of the LAST algorithm. The pseudocode in Listing 1 presents a high-level implementation of the algorithm. The algorithm is distributed; the tree is constructed starting from u , which selects one or several of its direct neighbors as child nodes in the tree. The child nodes in their turn choose their child nodes and so on.

Node u does not choose child nodes from all its one-hop neighbors, but only from those that are located closer to A than u itself. These neighbors form u 's *candidate set* S_{CN} , as seen in Figure 2. The goal is to choose a subset of nodes from the candidate set to create a tree structure that jointly optimizes the interference and the energy cost. This is done by computing the interference- and energy metrics for each possible selection from the candidate set.

Listing 1. An overview of the LAST algorithm

```

TREECONSTRUCTION( $A, u$ )
1  if this is the first received request
2    then if  $u$  is inside  $A$ 
3      then BROADCASTREQUEST( $A, u$ )
4      else  $childNodes \leftarrow$  FINDCHILDREN( $A, u$ )
5          for each  $node$  in  $childNodes$ 
6            do CONNECT( $u, node$ )
7            TREECONSTRUCTION( $A_{sub}, node$ )

FINDCHILDREN( $A, u$ )
1  for each  $S \in S_{CN}, |S| \leq 2$ 
2    do for each possible division of  $A$  into subareas
3      do calculate  $EC(S)$ 
4          calculate  $IC(S)$ 
5  return  $S$  with minimum  $IC(S) + c * EC(S)$ 

RECEIVEREQUEST( $A, u$ )
1   $v \leftarrow currentNode$ 
2  if  $v$  is inside  $A$ 
3    then if this is the first received request
4      then CONNECT( $u, v$ )
5      BROADCASTREQUEST( $A, v$ )

```

Selecting child nodes. Assume that we, given the node u and the area A , have a subset $S \in S_{CN}$. Each node $v \in S$ is assigned to create a tree that covers the region A_v , which is a subarea of A . If the subset only contains one node, $A_v = A$. The interference cost of S is defined to be the total coverage of the edges from u to all nodes in S , as stated in (1):

$$IC = \sum_{v \in S} \zeta(u, v) \quad (4)$$

The energy cost is the estimated total cost for each node v in S to cover its assigned subarea A_v , in addition to the energy cost to connect u with all nodes in S :

$$EC = \sum_{v \in S} c(u, v) + EC_v(A_v) \quad (5)$$

The total cost of a given subset based on the amount of interference and the estimated energy cost is

$$IC + \beta * EC \quad (6)$$

where the parameter β decides the relative importance of the interference and energy cost factors. The subset with the lowest cost is chosen by u , and the tree construction request is forwarded to those nodes.

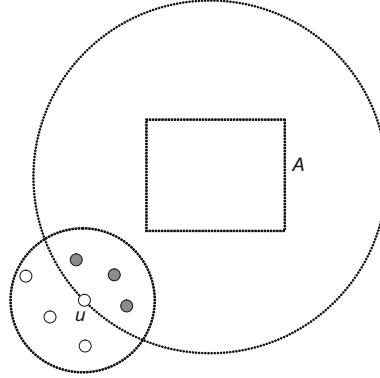


Fig. 2. The one-hop neighbors of u that are shaded are closer to the center point of A than u itself. They are therefore part of u 's candidate set.

The chosen subset is the one that combines low interference with low energy cost. In some cases, these objectives may be contradictory: the best option to reduce the interference might be to have paths that take detours between the sender and the destination. However, this will lead to energy costs that are higher than if the energy-optimal path had been used. Depending on the scenario, the “value” of a decrease in energy cost in terms of interference will be different, and the factor c can be changed to reflect that.

Branching decision. If a subset with only one node is selected, that node is assigned to cover the entire area A . It is also possible for u to select several nodes simultaneously from S_{CN} , and assign each of them to cover a subsection of A , so that all the nodes together are responsible for covering the entire area A . As described in more detail in the next section, A can be divided either horizontally or vertically, and the subareas are always of equal size. This is illustrated in Figure 3. Each node in the subset becomes the starting point of a branch, and is assigned to cover one such subarea.

Eventually, the LAST process will reach node v that is inside the particular subarea that is to be covered. (An example of this can be seen in the figure, where v is inside A_1^1 .) In that case, the algorithm will switch to the *broadcast mode*: the node v forms a connection to all its one-hop neighbors inside the area by broadcasting a request, which is forwarded by all the one-hop neighbors to their neighbors. If a node is reached by requests from several neighbors, it forms a connection to the node from which the first request is received. This is repeated until the tree covers the entire subarea.

Large candidate sets means that node u has to evaluate a large number of potential subsets. In order to reduce the computational load on individual nodes, u only considers subsets containing at most two nodes. In other words, the tree created by the LAST algorithm can branch into at most two branches at any given node. While this means that larger subsets will not be considered at all,

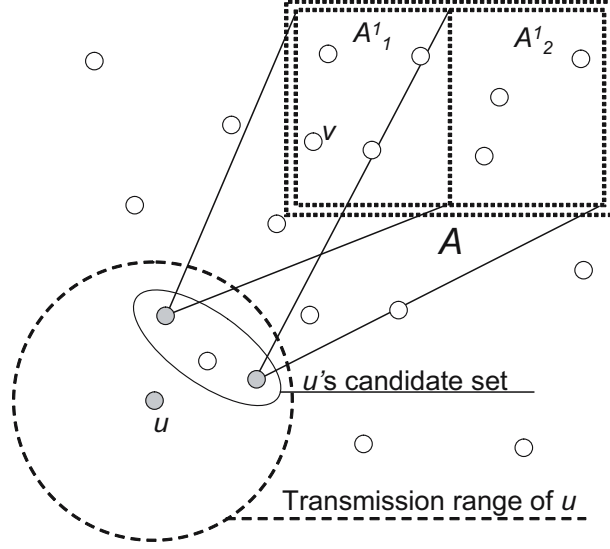


Fig. 3. Node u may use several nodes in its candidate set. Each of them will cover a subarea of region A .

the high number of branches would likely lead to a large amount of interference, which would make them unsuitable for the purpose of our algorithm.

For a candidate set S_{CN} , the number of unique non-empty subsets of a size between 1 and $|S_{CN}|$ is equal to $2^{|S_{CN}|} - 1$, which is no more than $2^\Delta - 1$, where Δ is the maximum degree of the network graph. However, the LAST algorithm is limited to no more than 2 branches at the same time, and correspondingly to subsets S such that $|S| \leq 2$. This means that the number of subsets considered by the algorithm is $O(\Delta^2)$.

Using brute-force calculation, we would have to compute the energy cost estimate and the edge coverage for all nodes in a particular subset in order to compute the energy cost and interference for that particular subset. Assuming a constant computation cost for calculating the estimated energy cost and the edge coverage for one node, the computation cost for one subset is constant. Thus, the total computation complexity for one node is $O(\Delta^2)$.

However, using this approach some of the calculations will be redundant. If node u is assigned to cover the region A , the two subareas that A will be divided into will be the same for any subset of S_{CN} of size 2. This means that for each node $v \in S_{CN}$, it is only necessary to estimate the energy cost to cover A and, at most, four different subareas of A (two for when A is divided horizontally and two more for when A is divided vertically). Since there will never be more than five different subareas to consider for each node, the total computation complexity for u to calculate the energy cost for all possible subsets will be $O(\Delta)$.

Division of target area. Depending on when and how the target area is divided, the topology and properties of the resulting tree will be completely different. This section discusses different aspects of choosing divisions that are more likely to result in trees with better performance.

The division of A can be done either horizontally or vertically, as is shown in Figure 4. Since different divisions will result in different trees, both the horizontal and the vertical divisions must be considered in order to evaluate all possible trees.

Since the areas and subareas will always be in form of rectangles, the communication overhead to describe their shape will be constant.

Packet losses during construction of the tree can lead to large parts of the tree never being constructed, which will result in only partial coverage of the target area. In order to reduce the effect of packet losses, the LAST algorithm creates subareas that overlap each other, as shown in Figure 5.

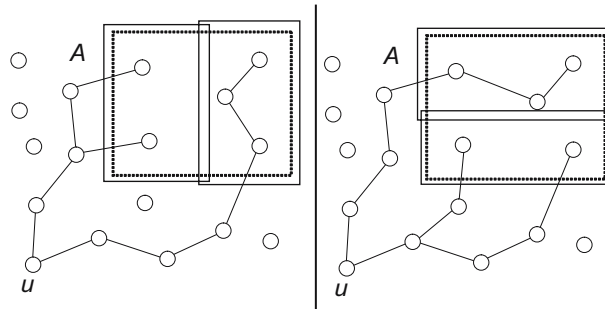


Fig. 4. In this case, node u has chosen a subset consisting of 2 candidate nodes, which means that A will be divided into two subareas. The resulting tree structures will look different, depending on whether A is divided horizontally or vertically.

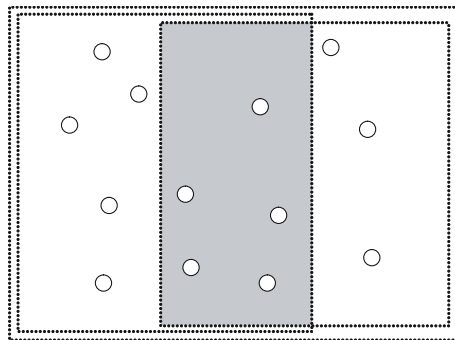


Fig. 5. The nodes in the gray area are covered by both the subareas. Thus, if the construction of a subtree covering one of the subareas fails, they will still be covered by the other subtree.

Minimum size of regions. As the tree constructed by the LAST algorithm branches, the regions to be covered by an individual branch become smaller. If the subareas are in the shape of thin rectangles, there is a high risk that the tree covering a subarea cannot connect all the nodes in the area without using nodes located outside of the area. This means that the energy cost of the tree will be higher, which is not desirable. A goal of the algorithm is therefore that neither the width nor the height of the subareas should be too small: an area A is not allowed to be divided so that the height or width of the resulting subarea falls below a given threshold.

No matter what the minimum size of a region is defined to be, it is not possible to guarantee that there exists a path between any two nodes in the region, if the path is required to be completely contained within the region. Figure 6 shows an example of this. Each of the nodes has a one-hop neighbor that is closer to the center node than they themselves are, so there are no stuck nodes. Still, the spiral topology can be of any size, which means that no region of any fixed size can be guaranteed to contain such a topology.

Target points. It is not certain that the best way, with respect to the energy cost, to construct a tree is to always aim for the center points of the target areas. Therefore, the LAST algorithm makes it possible to specify target points located anywhere in or on the edge of the areas.

When the LAST algorithm reaches a branching point, and the area to be covered is divided into two, the target points will be located on opposite corners along the split side that is facing the branching point. In the left part of Figure 4, the target points would be located at the two bottom left corners of the area A .

The method to estimate the energy cost described in 4.1 will generally result in trees that targets the center point c of A . The reason for this is that for a node

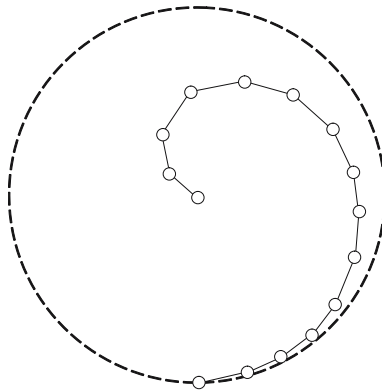


Fig. 6. Even though the network is assumed to have no stuck nodes, there is no limit on how much a path between two nodes might divert from the straight line between the nodes. This means that independent of the size and shape of a given area A , we cannot guarantee full connectivity within A using only the nodes inside A .

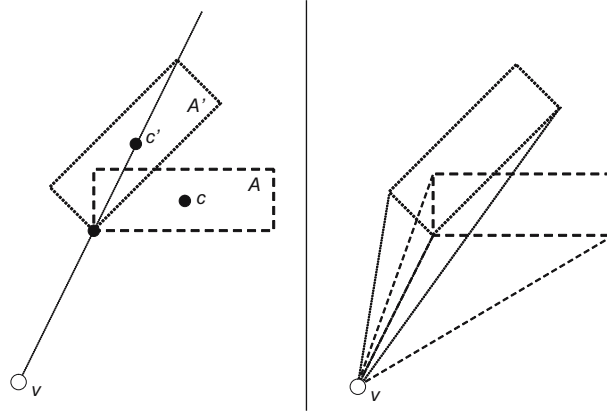


Fig. 7. When node v is assigned to enter the area A through point p_1 , A is rotated and transformed into A' . (Note that the line through u and p_1 also goes through the transformed center point c' .) This results in a smaller triangle area and accordingly a lower estimation of the energy cost.

u , choosing the nodes that are closest to c as child nodes will typically result in a lower estimated energy cost compared to choosing other nodes. In order to model that the tree is supposed to enter A near to one of its corners, the estimation method has to be modified: The tree should target the specific corner instead of the center point of A , and subsequently cover A starting from that corner. The estimation method that accomplishes this works similarly to the previous described method, with the only difference that the area A is repositioned before the circle sector is constructed.

When entering an area A in a specific point p_1 , the following computations are performed: the area A is rotated into A' so that a line that goes through v and p_1 also crosses the center point c' of A' . This can be seen at the left part of Figure 7. The actual location of A is indicated by a dashed rectangle, and the rotated position by a dotted rectangle. Note that the modified (dotted) circle sector covers a smaller area than the original (dashed) circle sector. This means that the estimation method indicates that a tree structure that enters A near p_1 will require less energy than a tree structure that would aim for A in general.

5 Discussion and Future Work

5.1 Degree of Overlap

In the case of packet losses, the construction of a branch might be interrupted, and a portion of the area A may not be covered. In order to remedy this, the subareas can overlap each other in order to provide some redundancy, as in Figure 5. This can lead to nodes being covered by several different branches, in which case a node only replies to the first branch it is contacted by.

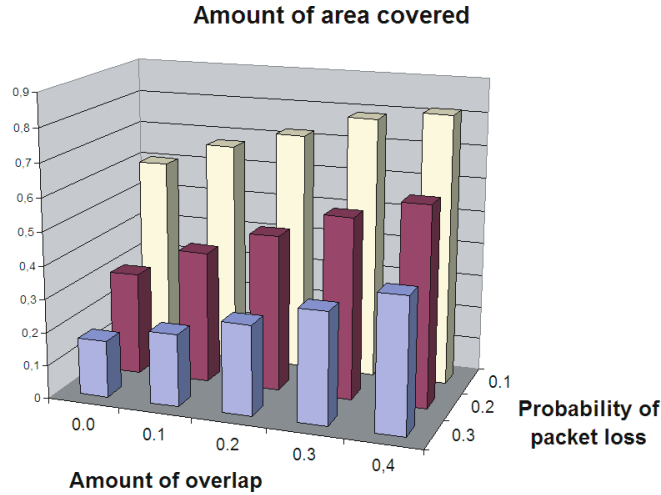


Fig. 8. Increasing the amount of overlap leads to a larger part of the area being covered

Determining the degree of overlap between different areas is a tradeoff between robustness and effectiveness. If an area is covered by several branches, the resulting tree structure will be ineffective with respect to both energy cost and interference. However, the impact of packet losses will be reduced.

In order to evaluate the results for different degrees of overlap, simulations were performed for different values of amount of overlap and probability of packet loss. For the purpose of the simulations, we assumed that the LAST algorithm would always create a tree that branches at four levels between the root node and the target area. This means that the tree has a total of $2^4 = 16$ leaves, where each leaf is responsible for covering a subarea of the target area.

Between each pair of branching points in the tree, there was a fixed probability of packet loss during the construction of the tree. When a packet is lost at some point, the construction of the entire subtree below that point is terminated.

In the simulations, the original area to be covered was always quadratical. At branching points, the area was always divided so that the resulting subareas would be as close to quadratical as possible. In other words, if the height was bigger than the width the area was divided horizontally, and vice versa.

The amount of overlap, as well as the probability of packet loss between two branching points, were varied in the simulations. For each possible combination, 500 simulation runs were performed. The results can be seen in Figure 8.

5.2 Implementation Issues

The case of non-uniform node distributions. Our estimation of the energy cost to cover an area from a specified node assumes that the sensor nodes are uniformly distributed over the area. If this is not the case, comparing different estimated costs may give incorrect results. However, to ensure a correct energy

estimation it would be necessary to collect node density information over the entire network, which would be costly in terms of energy spent by the nodes.

Load balancing. In order to avoid overloading links and nodes near the root of the tree, it is not possible to use the links near the branches of the tree to their full capacity. As the root node starts the tree construction, it specifies the maximum possible bandwidth b . At the first branching point, each of the two branches can only transmit information at the maximum rate of $\frac{b}{2}$. At the next branching point, the maximum bandwidth is limited to $\frac{b}{4}$. This is repeated all the way to the leaves of the tree.

Branching penalties. If the sensor network is very dense, a higher number of branches may lead to increased interference that is not considered by our interference metric. In such a scenario, it might be desirable to avoid excessive branching in the tree structure even if our metric indicates that it is the best choice.

This can be controlled by adding an extra "penalty" value to (6) for all subsets with size 2, so that they have to perform significantly better than the subsets with only one member in order to be chosen.

5.3 Correctness of the Algorithm

The LAST algorithm is a recursive algorithm, with the base case that a branch has reached a node inside the subarea to be covered. If our assumptions of connectivity of the network holds, it can be shown that the following broadcast phase is guaranteed to eventually reach all the nodes in the subarea. For the algorithm to be proven correct, it has to be shown that it eventually terminates, and that the entire region A will be covered at that time. In this section, we present a simplified sketch of the proof.

We assume that, for any node u and any given point in the plane q outside that node's transmission range, that node has a neighbor that is closer to q than u itself. Thus, it can be shown that each node u in the network has a non-empty candidate set consisting of nodes that is closer to the region A than u itself. For every step in the tree construction algorithm, the node or nodes selected will be closer to the area to be covered than their parent node. Eventually, a node inside the area will be selected, and the base case has been reached.

If, at some point, two nodes in the candidate set are selected, and the tree splits into two branches, the task to cover the area A is divided over the branches so that the entire area will still be covered by all the branches taken together. The only way in which the algorithm will fail to create a tree that covers the entire specified region is if we take data losses into account. This has been discussed in more detail in Section 5.1.

5.4 Implementation

The LAST algorithm is currently being implemented in TinyOS operating system for sensor nodes. We extend the functionality of a combination of TinyLUNAR [6] and GPSR [4] protocols. TinyLUNAR is the adopted to the specifics

of sensor networks connection oriented routing scheme originally developed for mobile wireless ad hoc networks. The major property of LUNAR is simplicity of implementation in comparison to other protocols developed for MANETs. The most illustrative performance characteristic of TinyLUNAR is that it introduces only one byte overhead per data packet when forwarding on a multihop path.

In its turn GPSR [4], the Greedy Perimeter Stateless Routing is a geographic routing protocol proposed for wireless ad hoc and sensor networks that can be used to route data packets between any pair of nodes identified by their geographic coordinates. GPSR assumes that every sensor node is aware of its own location and the locations of its neighbors.

For the implementation of the LAST algorithm we combine the best features of the two protocols. Instead of broadcasting the route request messages in TinyLUNAR we carry them via GPSR to the target geographic region. Each node receiving the RREQ message from GPSR establishes a backward label switching path. Upon the completion of the route establishment phase the messages are sent using the efficient forwarding mechanism of TinyLUNAR by this avoiding the overhead that otherwise would be introduced by packet forwarding directly over GPSR.

6 Conclusions

We presented a multicast tree formation algorithm for short-range wireless networks. Unlike previous algorithms that we are aware of, the LAST structure covers all the nodes in a geographic area rather than specific individual nodes. Previous algorithms have taken energy cost into account in order to prolong the lifetime of the network. In comparison to the existing approaches the LAST algorithm introduces interference as a metric when building multicast trees.

References

1. Burkhart, M., von Rickenbach, P., Wattenhofer, R., Zollinger, A.: Does topology control reduce interference? In: Proceedings of the 5th ACM Int. Symposium on Mobile Ad-hoc Networking and Computing (MobiHoc), pp. 9–19 (2004)
2. Frey, H., Ingelrest, F., Simplot-Ryl, D.: Localized minimum spanning tree based multicast routing with energy-efficient guaranteed delivery in ad hoc and sensor networks. Technical report, University of Southern Denmark, Ecole Polytechnique Federale de Lausanne, Universite des Sciences et Technologie de Lille (2007)
3. Johansson, T., Carr-Motyckova, L.: Reducing interference in ad hoc networks through topology control. In: Proceedings of The 2005 Joint Workshop on Foundations of Mobile Computing, pp. 17–23 (2005)
4. Karp, B., Kung, H.T.: Gpsr: greedy perimeter stateless routing for wireless networks. In: MobiCom 2000: Proceedings of the 6th annual international conference on Mobile computing and networking, pp. 243–254. ACM, New York (2000)
5. Koutsonikolas, D., Das, S., Hu, Y.C., Stojmenovic, I.: Hierarchical geographic multicast routing for wireless sensor networks. In: International Conference on Sensor Technologies and Applications, 2007. SensorComm 2007, October 14–20, pp. 347–354 (2007)

6. Osipov, E.: tinyLUNAR: One-byte multihop communications through hybrid routing in wireless sensor networks. In: Koucheryavy, Y., Harju, J., Sayenko, A. (eds.) NEW2AN 2007. LNCS, vol. 4712, pp. 379–392. Springer, Heidelberg (2007)
7. Sanchez, J., Ruiz, P., Stojmenovic, I.: Gmr: Geographic multicast routing for wireless sensor networks. In: 3rd Annual IEEE Communications Society on Sensor and Ad Hoc Communications and Networks, 2006. SECON 2006 , vol. 1, pp. 20–29 (September 2006)
8. von Rickenbach, P., Schmid, S., Wattenhofer, R., Zollinger, A.: A robust interference model for wireless ad-hoc networks. In: Proceedings of the 19th IEEE International Parallel and Distributed Processing Symposium (IPDPS (April 2005)
9. Wu, S., Candan, K.S.: Gmp: Distributed geographic multicast routing in wireless sensor networks. In: ICDCS 2006: Proceedings of the 26th IEEE International Conference on Distributed Computing Systems, Washington, DC, USA, p. 49. IEEE Computer Society, Los Alamitos (2006)
10. Zeng, G., Wang, C., Xiao, L.: Grid multicast: an energy-efficient multicast algorithm for wireless sensor networks. In: Fourth International Conference on Networked Sensing Systems, 2007. INSS 2007, June 6-8, pp. 267–274 (2007)
11. Zhang, H., Arora, A., Choi, Y., Gouda, M.G.: Reliable bursty convergecast in wireless sensor networks. In: MobiHoc 2005: Proceedings of the 6th ACM international symposium on Mobile ad hoc networking and computing, pp. 266–276. ACM, New York (2005)