LSynD: Localized Synopsis Diffusion

Andreea Berfield, Panos K. Chrysanthis, Daniel Mossé
Department of Computer Science
University of Pittsburgh
{andreea, panos, mosse}@cs.pitt.edu

Abstract

Wireless sensor networks represent an extremely fast-growing emerging technology, but still suffer from several limitations. The state of the art in sensor networks focuses on optimizing the existing protocols to address the two main challenges affecting the sensors: failures and energy consumption. Our contributions in this paper include: analyzing most relevant protocols that attempt to address these two problems, presenting methods to achieve local reconstruction for a sensor network that uses multi-path routing and proposing a new protocol, called LSynD*, an extension of the Tributaries and Deltas approach. LSynD achieves a faster, more localized and energy efficient reconstruction than its predecessor protocols by creating multiple adaptive multi-path routing regions.*

Keywords: failures, energy saving, synopsis diffusion

1 Introduction

Sensors are small wireless devices used for collecting data from the physical environment in real-time for critical applications such as habitat monitoring and crisis management. Sensors are resource-constrained devices with limited transmission range. Furthermore, by being battery-operated, sensors have limited power that restricts the amount of processing and communication that they can perform before they become inactive. In order to overcome these limitations, sensors usually organize themselves into a hierarchical topology, called a *routing tree*. Within the routing tree, each sensor picks another sensor within its transmission range to be its parent. Different criteria can be used for selecting the best choice for the parent node [19, 20]. Once this child-parent relationship has been established, sensors can collect data from their children, aggregate it with their own readings and forward it to their respective parents [12, 21, 19]. The root of the tree is a special node, able to communicate the readings collected from the network to the end user.

Failures represent another major challenge to any real-life deployment of sensors, because they can significantly decrease the quality of the final response or even leave the user unable to access part of the sensor network. Failures occur due to two main reasons, namely, (a) exposure to the elements of the nature, and (b) the limited energy capacity of the sensors and the subsequent inability to access and recharge the sensors. A solution to the latter problem is to use intelligent protocols that conserve sufficient sensor's energy to provide high quality responses for the required duration of the application [18, 2]. On the other hand, irrespective of the cause, failures are, in general, unpredictable and the only way to handle them is through some form of redundancy that either masks faults or avoids failures [3, 10, 7, 17, 11].

Current fault handling approaches can be broadly classified as *single path redundancy* or *multi-path redundancy* schemes, depending on the way they route information in the network. In single-path redundancy schemes, data is sent every round on a different path to increase the chance of the user obtaining a result (e.g., [6, 21, 1]). In contrast, multiple path redundancy schemes send partial data on different paths simultaneously at every round and gathers the data at the root (e.g., [12, 5, 15]).

Among the most successful fault-tolerant schemes are Sketches [5] and Synopsis Diffusion (SynD) [15], which use duplicate-insensitive multi-path forwarding. These two protocols are very similar, both using approximation methods and usually providing a highly-desired low error rate. Unfortunately, they provide approximate answers even in the *absence* of failures. Further, these schemes increase energy consumption for most of the aggregate functions, which can be solved with RideSharing approaches [9]. Another solution, Tributaries and Deltas [13], was devised to address the drawbacks of and improves on SynD by creating a hybrid protocol that employs a simple routing tree if there is no failure and repairs failures by transforming the tree to a multi-path redundant network as in SynD. In this manner, if there is no failure, the reported result suffers no approximation. However, Tributaries and Deltas fail to

Proceedings of the 10th IEEE International Symposium on
Object and Component-Oriented Real-Time Distributed Computing (ISORC'07)
0-7695-2765-5/07 $20.00 © 2007 IEEE

decrease energy consumption and is slow in responding to failures because of the global nature of the reconstruction of the multi-path sensor network.

In this paper, we focus on the localization of failures and recovery in multi-path routing schemes. Specifically, our contribution in this paper is twofold.

1. We discuss methods to achieve local reconstruction of the routing tree after failures for a sensor network that uses multi-path routing. These are based on the principle of partitioning across several dimensions.

2. We devise a new protocol, called *LSynD* (**L**ocalized **Syn**opsis **D**iffusion protocol), as an extension of the Tributaries and Deltas approach. LSynD is faster, more localized and more energy efficient than its predecessor protocols because it adaptively partitions a routing tree into multiple multi-path subtrees.

The remainder of this paper is organized as follows: in the next section we describe the protocols proposed in Tributaries and Deltas in detail and illustrate their shortcomings. We also present different approaches to achieve local reconstruction of the routing trees and possible improvement opportunities. We describe our proposed LSynD protocol in Section 3 and its performance evaluation in Section 4. We conclude in Section 5.

2 Tributaries and Deltas: Challenges and Opportunities

In this section we describe the scheme on which we base our work, namely, Tributaries and Deltas (TD) [13]. Subsection 2.1 introduces the system model and basic concepts. Subsection 2.2 presents TD's merits and discusses its potential problems. Subsection 2.3 continues the discussion with the introduction of the opportunities for improvement for the localized routing tree reconstruction.

2.1 System Model and Synopsis Diffusion

In this paper we assume a sensor network consisting of nodes with typical communication and computation capabilities that process continuous queries (and possibly perform in-network aggregation). The user interfaces with the network through a special node, called *base station* (BS).

In order to distribute queries to the sensor network and collect responses from it, sensors are organized in a hierarchical topology such as *routing tree* or *concentric rings*. In both cases, the nodes are divided into levels or rings according to their hop count from the BS, which forms the root of the hierarchy at level 0.

The *routing tree* topology is based on a child-parent relationship, where the parents are located one level closer to the BS than the children. A sensor node aggregates and forwards data from its children to its parent. By aggregating the data, a sensor node can reduce its power consumption by transmitting only one aggregate value for a series of results.

The *concentric rings* topology (used in [15]) is similar to the routing tree topology in both the way they are constructed and the way data is propagated towards the BS. In this topology, each node broadcasts its data to multiple neighbors and only those neighbors on a lower or inner ring (closer to the root) are responsible to propagate this data towards the BS (the root). Thus, the multi-path propagation is performed in a level by level fashion towards the BS, as in single-path propagation in routing trees.

The concentric rings topology supporting multi-path propagation is more robust than the routing tree one as long as no aggregation or duplicate-insensitive aggregation is involved such as MAX or MIN. *Synopsis diffusion* (SynD) [15], is a duplicate-insensitive in-network aggregation framework that decouples aggregation from message routing, thus enabling arbitrary multi-path routing schemes, adaptive to the network conditions.

In the SynD in-network aggregation framework, the partial result at a node is represented as a *synopsis*, that is, a small digest of the data in the form of hash tables. A node updates its synopsis by fusing its result with all synopses contained in messages transmitted from nodes in its broadcast range situated on a higher/outer ring. If needed, nodes are able to change rings in order to deal with failures. The value of a synopsis is approximated with the following formula: $2^{z_{avg_i}} * m/0.77351$, where m is the number of hash tables used, z_i is the position of the least significant 0 in hash table i and z_{avg_i} is the average of all z_i.

Clearly, the use of order- and duplicate-insensitive (ODI) synopses as part of the messages exchanged between sensor nodes improves the robustness of the concentric rings topology. However, this improvement in robustness comes at the price of having approximate answers even in the absence of failures and larger messages overall.

2.2 Tributaries and Deltas

The Tributaries and Deltas (TD) [13] approach attempts to address the drawbacks of SynD by creating a hybrid protocol that employs a simple routing tree if there is no failure and repairs failures by transforming the tree into a multi-path concentric rings network as in SynD.

Specifically, TD proposes a hybrid of single-path region, also called *Tree, Tributary* or *T region*, and multi-path region, also called *Delta, D or* M region[1](Figure 1).

[1]We adopt the notation D-nodes or D-region for compatibility with the Delta naming.

Proceedings of the 10th IEEE International Symposium on
Object and Component-Oriented Real-Time Distributed Computing (ISORC'07)
0-7695-2765-5/07 $20.00 © 2007 IEEE

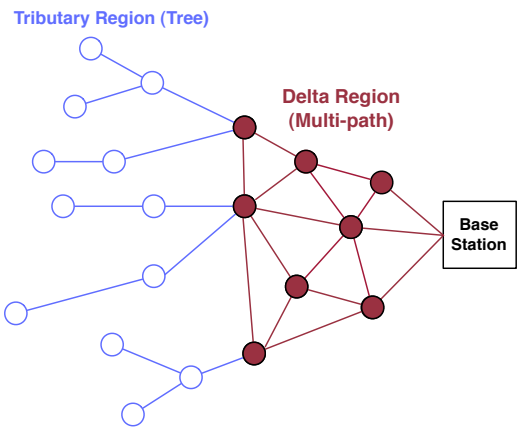

Figure 1. Tributaries and Deltas

The D-region looks like a delta, starting at the BS and expands d levels down the routing tree (RT). All sensor nodes in the D region (i.e., on the first d levels in the RT) will use multi-path routing, while the rest of the nodes (i.e., from the leaves up to level $d+1$) in the tree will use single-path routing. As in SynD, all the D-nodes collect messages originated from a T-node or D-node on an outer ring or higher level[2], fuse them with their own values and broadcast a new synopsis. The BS combines all the received values into a single final result.

Periodically, based on state information collected from the network, the BS decides to increase/decrease the delta region and takes the necessary steps for restructuring the sensor network accordingly. These steps instruct nodes on a specific level to switch from being D-nodes to become T-nodes and vice versa. Based on the criteria and the steps taken, TDs are distinguished into two basic strategies: *Strategy TD-Coarse* and *Strategy TD*.

In *Strategy TD-Coarse*, nodes collect information about how many nodes contributed to the reported result and propagate this information towards the BS. The strategy works by expanding/shrinking the delta region by one level, until the count of participating nodes contributing to the result being sent falls below/above a threshold percentage of the total nodes in the network. Because it periodically changes the delta region only by one level, this protocol does not adapt well to different conditions in different parts of the network. Depending on the shape of the RT, it may also take a long time to increase the delta region sufficiently.

Strategy TD tries to overcome the problem of expanding D-regions, one level at a time, by working at a finer granularity. Each node sends additional information to its parent about the number of nodes in its subtree that did not contribute to its partial result. If we consider the sensor network as a directed graph, a D-node is considered *switchable* (to a T-node) if all its incoming edges are T edges or it has no incoming edges. Under Strategy TD, the BS switches all children of switchable D-nodes belonging to a subtree that has maximum/minimum nodes not contributing.

Strategy TD allows a more directed routing tree reconstruction, but it does not reconstruct all the affected areas, so it is not exactly a localized reconstruction. The authors mention as future work other heuristics that can be used as possible improvements to Strategy TD: using max/2 or top-k values instead of max. Even with these proposed heuristics, the protocol does not cover all the affected areas that need to be reconstructed. For example, the top-k or max heuristics might unnecessarily switch too many nodes from T-region to D-region.

Both Strategy TD-Coarse and Strategy TD start from a very innovative underlying idea: a hybrid single and multiple path sensor network that has the advantage of both approaches, namely good response in the presence of failures, lower or inexistent error during fault-free operation and lower energy consumption overall. Unfortunately, these schemes still suffer from some drawbacks: (a) the BS is a single point of failure and (b) the expansion and shrinkage of the delta region (similar with routing tree reconstruction) happens in a centralized fashion and during this period the user is unable to get results from the network. The performance of the schemes is highly dependent on the ratio of single-path to multi-path nodes existent in the network and on the location and localization of failures. For example, if some failures happen at the bottom levels in the routing tree, the entire tree (or a large part of it) must become multi-path to deal with these failures. The protocols may take time to sufficiently increase/decrease the delta region.

To illustrate this further, let us consider the example in Figure 2, which demonstrates how Strategy TD can make the wrong decision regarding what part of the tree to become a delta region at a specific point.

The figure depicts the effects of applying Strategy TD with the policy "maximum number of nodes not contributing". In this case, the left and right subtrees of the BS have 7 (out of 15) and 8 (out of 17) nodes, respectively, not contributing to the reported final result. The right subtree has a greater count, therefore all its nodes will switch to a delta region. However, in this subtree there is only a single alive node which is unable to communicate its sensed data. Hence, this switch is unnecessary. On the other hand, the left subtree has five nodes that would benefit from a delta region, but under this policy, these are not selected for the switch.

This example clearly illustrates the shortcomings of Strategy TD protocol with respect to both quality of data and efficiency in terms of time and energy. This also clearly illustrates the need for a scheme and a protocol that allow the creation of multiple multi-path regions in a localized

[2]Since a level in routing tree is functionally similar to a ring in the concentric rings topology, we will use the two terms interchangeably.

Proceedings of the 10th IEEE International Symposium on
Object and Component-Oriented Real-Time Distributed Computing (ISORC'07)
0-7695-2765-5/07 $20.00 © 2007 IEEE

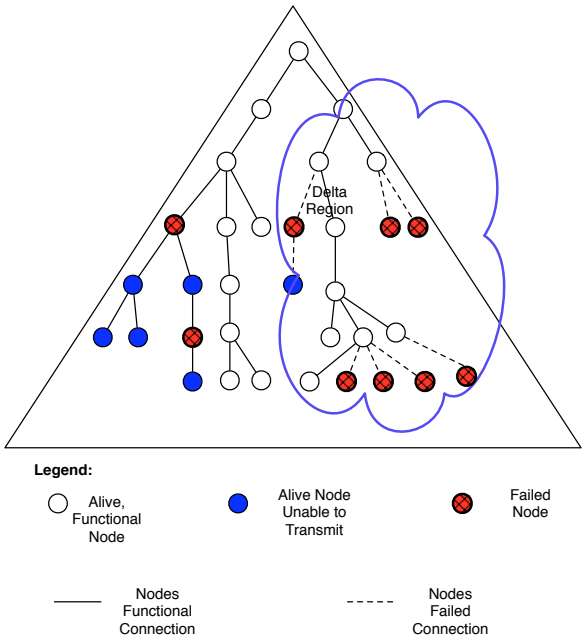

Figure 2. Routing Tree for Strategy TD

manner that copes with multiple failures, is faster and more energy efficient than TD strategies.

2.3 Localized Reconstruction

An underlying principle of partition is localization. Partitioning the sensor network into multiple routing trees has been strongly advocated for localization of failures by [8, 14, 16]. Such partitioning has additional advantages that affect both the quality of service and quality of data, including shorter routing trees, allowing for shorter sampling rates, and more balanced routing trees, that increase the life-span of the network.

Thus, one approach to localization in conjunction with multi-path routing as in SynD is the partition of a sensor network into multiple routing trees using multiple BSs. This scheme will give us more flexibility in managing the different multi-path regions. In order to apply the usage of synopsis from a single routing tree to multiple routing trees, one must ensure that results will not be infinitely propagated from one tree to the other. Also, one must ensure that the epoch consistency is respected. One solution is for sensors to consider only the messages broadcast from other sensors within their own routing tree and ignore any other messages.

This solution can be improved by allowing dynamic changes to the routing trees. As in SynD, whenever a node needs to change rings because it has no parent available on the upper ring, it moves one ring lower to connect via its siblings. In case this node is a leaf node, then it can choose to join any other neighboring routing tree and becomes a leaf node in that tree in order to retain or improve its level (i.e., move to a lower level/inner ring).

In the case of TDs, localization and partitioning can be done in the context of subtrees. Recall that the delta region always starts at the BS and may expand to the whole tree. This may cause much overhead, which can be avoided with localized D-regions. We propose that a tree be partitioned into subtrees (clusters), allowing multi-path transmissions within these subtrees; the roots of these subtrees (cluster heads) can correctly handle the duplicates (that is, these roots will aggregate and transmit a single value to their T-node parents). Clearly, this scheme gives more flexibility in creating and manipulating more localized delta regions within a tree. The only challenge is how to define these clusters and to efficiently (both in terms of energy and time) inform the sensors which cluster they belong to.

A simple solution is to define clusters in a static manner, before the network is operational. If the cluster head dies, then a new one must be elected and all the sensors in the cluster must be informed about it. As usual, the biggest problem with pre-defined static clusters is that it does not take into account the dynamicity of the network (it is hard to predict where sensors are going to fail and what shape the delta region should have).

Another option for static clustering is to pre-determine which sensors neighboring a different cluster to switch to what clusters; eventually all the sensors in the failed cluster will join another cluster. The main problem with this approach is that it would take a long time to switch a portion of a tree to belong to a neighboring tree.

It would be more useful to be able to define clusters dynamically, whenever needed and have delta regions tailored to the failed sections of the network. Our proposed protocol LSynD, discussed in the next section, both eliminates most of the drawbacks of Strategy TD protocol and also enables the creation of adaptive clusters.

3 Localized Synopsis Diffusion

In this section, we discuss in detail our *Localized Synopsis Diffusion (LSynD)* protocol whose versatility in dealing with failures in a sensor network is a result of its ability to create multiple, multi-path routing clusters on demand. Although, in this paper, we discuss LSynD in the context of a single routing tree and single BS, LSynD can be used in conjunction with multiple BSs that partition the sensor network into multiple trees as discussed above and in [14].

3.1 Overview

As mentioned above, Strategy TD and Strategy TD-Coarse both take into account a threshold of the percent-

Proceedings of the 10th IEEE International Symposium on
Object and Component-Oriented Real-Time Distributed Computing (ISORC'07)
0-7695-2765-5/07 $20.00 © 2007 IEEE

age of sensors participating to the final result. As we have shown with our example in Section 2.2, this threshold can be misleading, because it also takes into consideration failures of the leaf nodes. These nodes should not be a part of the total count of failed nodes (obtained by subtracting the number of participating nodes from the total number of sensors in the network). This is because increasing the delta region will not improve the quality of the final response. In our approach, rather than using a threshold based on the total number of nodes, we propose to compute this threshold based only on the number of *failed* nodes, but excluding the failed leaf nodes. We also propose a new protocol (described in Section 3.2) based on the concept of minimum and maximum levels in the routing tree that has failed nodes.

Note that strategy TD-Coarse does not take into account the localization of the failures and may have unnecessary energy overhead, expanding the delta region more than it is actually needed. Even though it improves on TD-Coarse, Strategy TD can also still be improved upon. LSynD tries to correct these problems and to answer questions like "What if there are only failures in different regions at the bottom of the routing tree(s)?" In this case, only these particular regions need to become delta regions, not the whole routing tree.

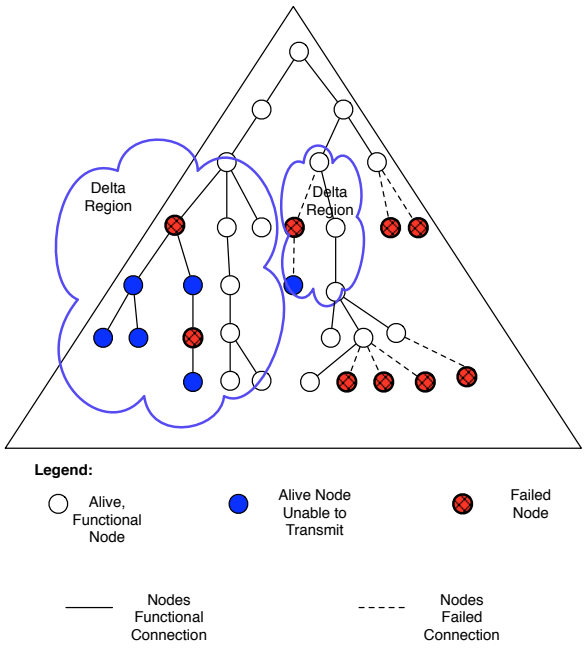

Figure 3. Routing Tree for LSynd

Figure 3 shows the behavior of LSynD for the same scenario as the one depicted in the Strategy TD example from Figure 2. In this example, LSynD manages to convert desired/targeted regions in the routing tree to delta regions, addressing the problem of all nodes that were previously alive, but unable to transmit. This is also accomplished without much overhead in terms of number of switched nodes, in addition to the nodes that were isolated before.

The delta regions can have different sizes and be rooted at different locations (nodes) in the sensor network. These special nodes (cluster heads) collect all the multi-path messages from their subtrees, aggregate them with their own values, transform and forward the partial result in a single-path manner. The sensors located at the bottom of the subtrees (still internal nodes in the RT routed at the BS, not leaves) have the opposite behavior, acting as a switch from the T-region to the D-region. They receive messages only from their children, perform aggregation, create a synopsis structure, and forward the partial result on multiple paths. The sensors that do not belong to any delta region route the information flow according to the traditional tree behavior. Clearly, for correctness, sensors do not aggregate the values received from other sensors belonging to a different D-region.

In order to be able to perform the switch from D to T values, the cluster heads need to have sufficient processing power to extract the T value from different synopsis, meaning to extract the position of the least significant 0 from the hash tables. The time necessary to produce this result is a function of the length of the synopsis, namely $O(logN)$, where N is the upper limit of the number sensors in the routing tree.

3.2 LSynD Protocol

The LSynD protocol works as follows. Initially, all the sensors belong to a single tree region, with a specific id. Each node has two variables, *MinLevel* and *MaxLevel*, denoting information about the levels closest and farthest from the BS that contains a sensor failure. These variables may not reflect the exact reality of the network, only the sensor's knowledge. Whenever a parent detects the failure of a child, the parent updates its MinLevel and MaxLevel variables and communicates them to its own parent. Each node receives values from all its children, so nodes can set these values based on information on descendants received from a child. The variables are updated in the following manner:

$$MinLevel = min(\infty, VC),$$

$$MaxLevel = max(-1, VC)$$

where ∞ and -1 are the default initialization values for the variables and VC are the values received from children. When a node notices that its parent is dead, it picks a different node (parent) to transmit to in the next round.

If the number of failed nodes in any of the BS's subtrees routed at its children exceeds a given threshold, then

Proceedings of the 10th IEEE International Symposium on
Object and Component-Oriented Real-Time Distributed Computing (ISORC'07)
0-7695-2765-5/07 $20.00 © 2007 IEEE

the BS informs its children to create or expand the delta region. This message propagates along the sub-trees and each node tests whether it needs to become a D node or not: all the nodes in between $MinLevel - 1$ and $MaxLevel + 1$ become delta nodes. The node located at $MinLevel - 1$ becomes the cluster head (in other words, the *D-T switch node* for its delta region). When the D-T switch node propagates the message asking for modifying the delta region to its children, it also includes its sensor ID as the cluster head ID. As mentioned above, each delta region must have a different ID, and thus this information is vital for the correct functioning of the protocol because nodes aggregate only information received from synopses transmitted by sensors belonging to the same delta region. Internal sensors that have no failures reported in their subtrees and are situated at $MaxLevel+1$ do not need to forward this message. This policy eliminates any unnecessary transmissions and saves energy. Any nodes isolated at the bottom of the subtree, that is, whose parents are dead, will eventually join another delta region; clearly, this is a shortcoming of any protocol, not only of our LSynD protocol.

The propagation of multi-path information up the routing tree stops at $MinLevel - 1$. The cluster head of the delta region transforms all the D values received into a single value and forwards it in a classic T manner to its parent.

In case a node, $node_i$, receives several delta region formation messages, it decides which delta region it belongs to, as follows. The node checks the sender of the message. If the sender is the node's active parent (from the list of possible parents, which are nodes on $level_i - 1$ and within $node_i$'s transmission range), then the node acts immediately, as described above (change the region type, etc). However, if the message is not from node's active parent, but the node believes that its parent is still alive, then $node_i$ waits a period of time. If the time interval expires and the node still has not heard from its active parent, it should check if it has received messages from other possible parents. If yes, it selects one of them (e.g., FIFO or energy-level policy) as active parent and joins that delta region. If no possible parents are alive, then the node should pick from its other choices (e.g., neighbor nodes, if it has any).

An interesting case is when a T-node notices that all its non-leaf children are dead. This node sets the MinLevel and MaxLevel variables pretending it is dead: in addition to MaxLevel (set knowing it has dead children), the node sets the MinLevel to its own level, instead to children's level. The node just pretends to be dead, but it still functions and the message broadcasted informs about the failed area. The reason for this fake death is that, by doing so, its delta region will include branches in the RT other than the one routed at this node and its grandchildren will have a chance to broadcast their values and contribute to the final result.

Our protocol is flexible in the conditions of D-region formation. Using a mechanism similar to the node with no alive children just described, non-leaf nodes with very low energy remaining (given threshold) can proactively "declare" themselves dead.

3.3 LSynD Optimization Discussion

LSynD protocol ensures that all the nodes in a given subtree between (MinLevel -1) and (MaxLevel +1) are D-nodes. We can further improve the reconstruction localization by taking more factors into consideration when deciding if a node is a D- or a T-node. Therefore, a node becomes a D-node if it satisfies both of the following conditions:

1. *it is on a level in between the specified interval* [*MinLevel -1, MaxLevel +1*]
2. *it has failed nodes in its subtree* OR *it has a failed neighbor* OR *it has a potential failed parent*

The protocol works as follows: all D-nodes forward the delta messages to their children. The T-nodes do not forward the message to join the delta region unless they are on a level smaller than $MinLevel - 1$. These extra conditions will allow construction of irregular-shaped delta regions, not only triangle delta regions[3]. Having some local multi-paths ensures that any additional failures will not have a large impact on the network, and will not leave these areas strained again.

Figure 4 illustrates the application of LSynD with the above optimization, on the RT we used in our examples in the previous section (Figure 3). The difference when compared to the base LSynD is that there are some nodes (gray nodes in the figure) that used to become part of the delta region with the base algorithm and with the optimization are no longer D-nodes. This, in turn, will make the network spend less energy overall.

4 Analytical Evaluation

The evaluation of a sensor network can be quantified from a few different perspectives: energy consumed, computing and time overhead and accuracy/quality of data (QoD) of the reported results. There are similarities in all three protocols, namely LSynD, Strategy TD and Strategy TD-Coarse. They all use the same size for the messages transmitted between sensors, which is 48-byte for the TinyDB system. This message size allows fits 40 32-bit "sum" synopses within a single message. Also, the monitoring cost for all schemes is the same, therefore the energy cost and QoD during normal processing (no failures) are the

[3] We call triangle delta regions the delta regions routed at any point in the RT situated at $MinLevel - 1$ and that extend to its subtree up to $MaxLevel + 1$; the shape of such delta region resembles a triangle.

Proceedings of the 10th IEEE International Symposium on
Object and Component-Oriented Real-Time Distributed Computing (ISORC'07)
0-7695-2765-5/07 $20.00 © 2007 IEEE

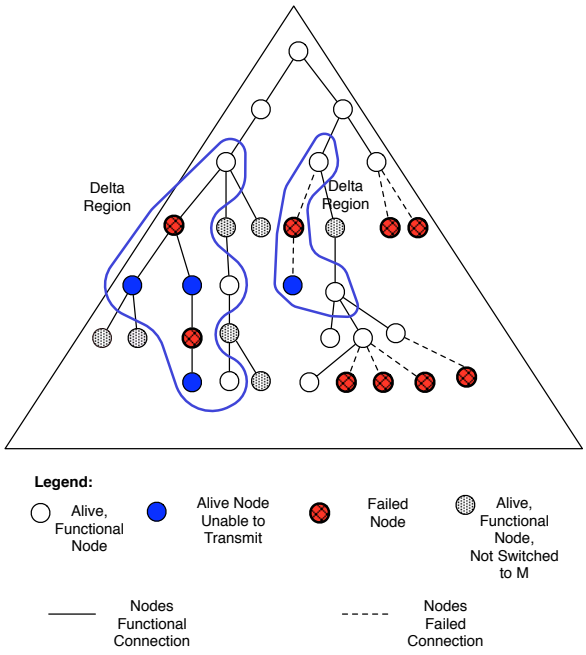

Figure 4. Routing Tree for LSynD with Optimization

same. Under LSynD and Tributaries and Deltas protocols, the sensor network starts with all nodes in T mode, so there is no approximation for the QoD in the absence of failures.

We have discussed that we need more capable sensors in the network to perform the cluster head function of transforming D values into a T value. While these cluster heads are only few nodes, all the extra D nodes under Strategy TD and Strategy TD-Coarse protocols also require some overhead for fusing their values into synopsis (computing[4], time, and energy overheads). Therefore, we will focus on the main difference among Tributaries and Deltas and LSynD protocols, namely how many nodes are switched to delta region.

The performance of all these protocols is highly dependent on the types of failures (we can consider sensor failures distributed in space as a whole area or as scattered point failures, and in time as simultaneous or incremental failures), their location and the percentage of D-nodes already in the sensor network. However, we were able to identify the best case and worst scenarios for comparing LSynD with Strategy TD and Strategy TD-Coarse for a grid topology.

The best case scenario for LSynD is when there are only failures at the bottom of the routing tree, as the following example illustrates. Let us assume that there are less than, say, 10% leaf nodes failed. In this case, LSynD will perform no reconstruction (delta region extension), because failed leaf nodes cannot be helped. Strategy TD-Coarse will perform total reconstruction and the number of D-nodes will be in the high 90% of the total number of nodes in the network (the entire routing tree is reconstructed even though nothing can be done for the failed leaf nodes). Strategy TD will reconstruct only one sub-tree and if we assume that the RT is balanced, then the number of D-nodes will most probably be in the range from 12.5% to 50% (or 1/8 to 1/2) of the total number of nodes in the network. This is because (for a grid topology) each sensor has at most eight neighbors and thus at most eight sub-trees (depending on the radio range, the node might have less neighbors). If the RT is not balanced, which is usually the case, then the percentage of nodes that will be switched to delta region can increase up to 75% or even high 90% for a BS with a single child (unlikely case, though).

The worst case for LSynD in the grid topology is when a whole region of the network rooted at the BS (and up to a level L) needs to be a delta region. In this case LSynD cannot add any improvements and will have the same performance as Strategy TD. In the extreme scenario, level L is the leaf level and the region switched to delta becomes the entire network.

In summary, the relative expected behavior of all protocols for a given grid network topology with 10% nodes failure is illustrated in Table 1.

Protocol	Percent of Delta Nodes		
	Best Case	Worst Case	Average Case
LSynD	0%	100%	50%
Strategy TD	90%	100%	95%
Strategy TD-Coarse	12.5% to 50%	100%	56.25% to 75%

Table 1. Protocols Evaluation

In conclusion, we expect that on average LSynD to perform better than the tributaries and deltas approaches. This analytical evaluation will be validated quantitatively through simulations in our future work.

5 Conclusions

A hybrid approach between the tree approach and multipath approach seems to be the ideal, providing both low overhead in terms of time and energy consumption. In this paper we analyzed protocols that deal with fault-tolerant in-network aggregation, noted some space for improvement, and proposed a means for more efficient and localized tree reconstruction in the event of faults. We have also proposed the LSynD protocol that enables localized reconstruction and fixes most of Strategy TD and Strategy TD-Coarse

[4] OR-ing the 40 synopses with its own synopsis.

Proceedings of the 10th IEEE International Symposium on
Object and Component-Oriented Real-Time Distributed Computing (ISORC'07)
0-7695-2765-5/07 $20.00 © 2007 IEEE

problems. Our protocol ignores leaf failures, it creates multiple and different size adaptive delta regions. The benefits from using LSynD, which avoids creating unnecessary D-nodes, include less overhead (with respect to nodes expanded into multi-path nodes) and faster reconstruction.

As part of the future work, we will construct a simulator that will allow us to quantify the performance improvements of LSynD over previous protocols.

Acknowledgments

This material is based on work supported by NSF Awards 0325353, 0524634, and 0549119.

References

[1] A. Berfield, P. K. Chrysanthis and A. Labrinidis. Efficient Handling of Sensor Failures. *The Third International Workshop on Data Management for Sensor Networks*, 2006.

[2] U. Cetintemel, A. Flinders and Y. Sun. Power-Efficient Data Dissemination in Wireless Sensor Networks. *The Third International ACM Workshop on Data Engineering for Wireless and Mobile Access*, 2003.

[3] S. Chessa and P. Santi. Comparison-Based System-Level Fault Diagnosis in Ad-Hoc Networks. *The 20th IEEE Symposium, Reliable Distributed Systems (RDS)*, 2001.

[4] W. Choi, S. K. Das, and K. Basu. Angle-based Dynamic Path Construction for Route Load Balancing in Wireless Sensor Networks. *Proceedings of IEEE Wireless Communications and Networking Conference (WCN)*, 2004.

[5] J. Considine, F. Li, G. Kollios and J. Byers. Approximate Aggregation Techniques for Sensor Databases. *The 20th International Conference on Data Engineering (ICDE)*, 2004.

[6] S. S. Dhillon and K. Chakrabarty. A Fault-Tolerant Approach to Sensor Deployment in Distributed Sensor Networks. *The 23rd Army Science Conference (ASC)*, 2002.

[7] H. Dubois-Ferriere and D. Estrin. Efficient and Practical Query Scoping in Sensor Networks. *The 1st IEEE International Conference on Mobile Ad-hoc and Sensor Systems (MASS)*, 2004.

[8] S. R. Gandham, M. Dawande, R. Prakash and S. Venkatesan. Energy Efficient Schemes for Wireless Sensor Networks with Multiple Mobile Base Stations. *IEEE Globecom*, 2003.

[9] S. Gobriel, S. Khattab, D. Mossé, J. Brustoloni and R. Melhem. RideSharing: Fault Tolerant Aggregation in Sensor Networks Using Corrective Actions. *The 3rd Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks (SECON)*, 2006.

[10] B. Krishnamachari and S. Iyengar. Distributed Bayesian Algorithms for Fault-Tolerant Event Region Detection in Wireless Sensor Networks. *IEEE Transactions on Computers (TC)*, 2004.

[11] F. Kuhn,T. Moscibroda and R. Wattenhofer. Fault-Tolerant Clustering in Ad Hoc and Sensor Networks. *The 26th International Conference on Distributed Computing Systems (ICDCS)*, 2006.

[12] S. Madden, M. J. Franklin, J. M. Hellerstein and W. Hong. TAG: A Tiny Aggregation Service for Ad-Hoc Sensor Networks. *The 5th Annual Symposium on Operating Systems Design and Implementation (OSDI)*, 2002.

[13] A. Manjhi, S. Nath, P. B. Gibbons. Tributaries and Deltas: Efficient and Robust Aggregation in Sensor Network Streams. *ACM Sigmod Conference*, 2005.

[14] A. Munteanu, J. Beaver, A. Labrinidis and P. K. Chrysanthis. Multiple Query Routing Trees in Sensor Networks. *The IASTED International Conference on Databases and Applications (DBA)*, 2005.

[15] S. Nath, P. B. Gibbons, S. Seshan and Z. R. Anderson. Synopsis diffusion for robust aggregation in sensor networks. *ACM SenSys*, 2004.

[16] E. I. Oyman and C. Ersoy. Multiple Sink Network Design Problem in Large Scale Wireless Sensor Networks (ICC). *IEEE International Conference on Communications*, 2004.

[17] M. A. Shah, J. M. Hellerstein and E. A. Brewer. Highly-Available, Fault-Tolerant, Parallel Dataflows. *ACM Sigmod Conference*, 2004.

[18] M. A. Sharaf, J. Beaver, A. Labrinidis, and P. K. Chrysanthis. TiNA: A Scheme for Temporal CoherencyAware in Network Aggregation. *Third International ACM Workshop on Data Engineering for Wireless and Mobile Access*, 2003.

[19] M. A. Sharaf, J. Beaver, A. Labrinidis, and P. K. Chrysanthis. Balancing Energy Efficiency and Quality of Aggregate Data in Sensor Networks. *The VLDB Journal*, 13(4), 2004.

[20] G. Santhanakrishnan, Q. Li, J. Beaver, P. Chrysanthis, A. Amer and A. Labrinidis. Multi-Criteria Routing in Pervasive Environment with Sensors. *The IEEE International Conference on Pervasive Services*, 2005.

[21] Y. Yao and J. Gehrke. Query Processing for Sensor Networks. *The First Biennial Conference on Innovative Data Systems Research (CIDR)*, 2003.

Proceedings of the 10th IEEE International Symposium on
Object and Component-Oriented Real-Time Distributed Computing (ISORC'07)
0-7695-2765-5/07 $20.00 © 2007 IEEE