On Multi-Path Aggregation of Duplicate-Sensitive Functions

Milton Armando Cunguara,Tomás António Mendes Oliveira e Silva
IEETA / University of Aveiro
Aveiro, Portugal
{milton.cunguara,tos}@ua.pt

Paulo Bacelar Reis Pedreiras
IT / University of Aveiro
Aveiro, Portugal
pbrp@ua.pt

***Abstract*—Previous decades brought about a revolution in radio and microprocessor technology that made possible a plethora of new applications. In particular, the possibility of using many inexpensive sensor nodes interconnected by wireless networks (WSN) for a number of ends, such as pollution monitoring and defense, draw the attention of the research community. WSN are usually heavily resource-constrained. Of particular relevance is energy, since in many applications nodes should operate during long periods from batteries. The literature on this topic reveals many techniques to improve energy efficiency, one of them being the use of network aggregation. However, the problem of aggregation of duplicate sensitive summaries (e.g. sum, average, histogram, etc.) in multi-path routing networks is not fully resolved. This paper addresses this problem by sending redundant aggregated information through different paths, so data can be reconstruct to obtain the exact summary, provided that there is at least one feasible path. Two algorithms are presented, one better suited for networks dominated by link errors and another suited to networks where the predominant error source is node failures. The algorithms are light during normal network operation, with the most intensive processing performed during the initialization phase. The approach presented herein outperform previous solutions found in the literature in two key aspects: complete topology independence and aggregation depth independence.**

***Index Terms*—Wireless Sensor Networks, Aggregation, Multipath, Spatial Query**

I. Introduction

Wireless sensor networks (WSN) have recently emerged as a synergy of two related technologies, i.e. radio and microprocessors technology. Both of them had exponential improvements in size, cost and functionality in recent years, opening the door to applications so diverse as air monitoring, forest fire detection, structural monitoring, water monitoring, etc. Furthermore, such improvements are expected to keep the pace in the near future, thus WSN should become even more prevalent.

Nonetheless, there are a number of aspects that need to be addressed in order to make WSN reach their full potential. One of the key aspects is energy efficiency, since in many application domains node's energy source are batteries and WSN should operate during long periods, with low or no maintenance at all. This aspect turned energy conservation, i.e. devising mechanisms to maximize the lifetime of the network, one of the most studied aspect of WSN.

This work was partially supported by the Portuguese Government through FCT - "Fundao para a Ciłncia e a Tecnologia" in the scope of project Serv-CPS -PTDC/EEA-AUT/122362/2010 and Ph.D. grant - SFRH/BD/62178/2009

Techniques proposed to reduce nodes' energy consumption include the use of network aggregation, the use of cluster heads (with or without cluster head rotation), exploitation of the inherent spatial correlation of readings among neighbor nodes, variation of transmission energy, sleeping during long periods of inactivity, exploitation of temporal correlation of the signals (data caching, estimation, system identification and so on), among others.

One approach to data transmission from common nodes to the root is to have all nodes transmit their data integrally to the root and at the root perform the computations on the collected data. However, this approach is energy-wise inefficient due to the rather high number of messages that are sent. A more efficient approach is the gradual aggregation of values as they are transmitted upstream, which is called **in-network aggregation**.

On the other hand, WSN links are *fragile*. Particularly, they can be temporarily unavailable, are subject to relatively high error rates or be asymmetric, among other issues. Hence, the use of multi-path routing has been proposed [1]–[4] to lessen its effects.

However, the use of multi-path routing may cause errors in the aggregation of duplicate-sensitive functions. A function is said to be duplicate-insensitive if its result does not change upon the introduction of a duplicate argument. For example, $\min$ and $\max$ function are duplicate-insensitive. A function is said to be duplicate-sensitive if its output changes with the addition of an argument, even if it is a duplicate. For example the sum function, in which introducing a non-zero argument more than once causes different results, is duplicate-sensitive.

Many attempts have been made to solve this issue, for example, by giving approximate answers, by forcing single path routes, by searching for aggregate-insensitive versions or by decomposing these functions in a series of duplicate-insensitive functions and then query the WSN for each of the new functions, among others.

None of the already introduced solutions satisfies the initial goals, since all of them either have an error by nature (approximations based approaches) which contradicts the goals of multi-path, or use single routes, which does not offer any redundancy, or considerable increase the number of messages that are required to compute the aggregate (alternative functions approaches), thus defeating the original purpose of the aggregation, which is to reduce the number of messages in order to save energy.

This paper addresses this problem by sending redundant aggregated information, so that data can be reconstructed to obtain the exact summary, provided that there is at least one possible path. Two algorithms are presented, one better suited for networks dominated by link errors and another suited to networks where the predominant error source is node failures. The algorithms are light during normal network operation, with the most intensive processing performed during the initialization phase. The approach presented herein outperforms previous solutions found in the literature in two key aspects: complete topology independence and aggregation depth independence.

The remaining of this paper is organized as follows. Section II presents an overview of the related work. Section III presents the methodology proposed in this paper to carry out the aggregate of duplicate-sensitive data in multi-path networks. Section IV presents simulation results carried out to assess the correctness of the algorithms and to evaluate its performance. Finally, section V concludes the paper.

II. Related Work

The field of WSN has a vast body of knowledge, too vast to be covered in the related work section of a single paper as can be seen in [5]. Therefore, this section is focused only in contributions closely related to problem addressed in this paper.

The use of aggregation in WSN excels in metrics such as energy expenditure and network lifetime, as shown in [6]. Furthermore, in the same reference it is also shown that the denser the networks the higher the benefits of aggregation. Similar results were reported in [7]. [8] presents a meta level view of aggregation and argues for a co-design of the integrating parts of the network.

Watfa *et al* [9] build an index tree similar to the SRT of TinyDB [10] (broadcast level and $\text{level} = \min(\text{level}) + 1$). The authors used an index table to decide the aggregation value, plus a common value agreement, which is a value that nodes with the same parent are supposed to verify $\text{abs}(\text{node}_{\text{reading}} - \text{cv}) < \delta$. cv is computed as the average of the children reading and is sent back to them. If a children verifies the last condition then it does not send its data. $\max / \min$ are also based on cv, thus may lead to erroneous values. In fact, this aspect is common to approaches that use a cluster head and spatial correlation. Nodes with more than one parent alternate between sending messages to each of its parents.

Liao *et al* [11] leverages the (ant) bio-inspired path finding algorithm to build a routing tree. In this algorithm, all source nodes explore all the paths to the sink, leaving a certain amount of pheromone at each link. The amount of pheromone left on each link of each path from a source to the sink is a function of several factors (e.g. path size). The amount of pheromone on each link is the sum of the amount left by on each paths that goes through the link. The higher a paths' pheromone levels, the more likely it will be active. However, this algorithm is not light. A similar approach is pursued in [12] and in references therein.

Ganesan *et al* [13] use a wavelet based approximate aggregation technique to store the results of several queries at different network hierarchies. Nodes at different hierarchies store results with different precisions. Whenever a query with a certain precision is done, it drills down the network until find a node with precision enough to answer it. Due to the high volume of data generated by this approach, an aging scheme was employed.

Considine *et al* [1] present the FM-SKETCH (introduced by Flajolet and Martin), which approximate the number of distinct elements of a superset by seeing only its initial part and such sketch are used to estimate the number of distinct nodes in a network — approximate count summary — which is well suited for multi-path networks. The paper generalizes FM-SKETCH to approximate sum summaries by having each node producing val (val is the value that the node sensed) distinct elements into the superset. Evidently, the number of distinct elements of this superset is equal to the sum summary. The sum summary is approximated by using the FM-SKETCH on the superset. A number of optimizations are provided. All the same, this approach that supposedly reduced the amount of data by computing approximates, actually for typical relative errors (0.85-0.95), it uses about the same amount of memory as traditional exact summaries. Therefore, the use of exact aggregation techniques with a data-caching or send-on-delta can be discarded. The authors compared their approach to TAG and to the LIST approach, i.e. each node sends the value of the aggregate and the set of nodes used to compute the aggregate, which is similar to the approach provided herein, however, their LIST approach is considerable less optimal. First, nodes always send a huge set, second the node that is receiving may not be capable of finding a way to aggregate the received data due to overlaps.

A Sparse aggregation scenario is studied in [14], by exploring dense WSNs with a small number of hotspots. A mechanism to find suitable routes was also presented. Other types of approximate aggregations have been proposed as is the case of quantile tracking [15], [16], top-k estimation [17], [18] though the later group tend to be more exact. [19] is an example in which an approximation of an histogram is used to compute queries in WSN, however, the algorithm used is exact if the histogram is exact, but the authors do not provide any algorithm to compute exact histograms.

In [20] and later at [21] the authors use a hybrid periodic$\backslash$asynchronous model, in which data is sent periodically, however, rapid transition are responded to by immediately sending data asynchronously. Nevertheless, the authors did not provide any difference between this paradigm and event triggered transmission with a periodic *I'am alive* messaging. The asynchronous traffic is controlled using filters. Whenever a value is within the filter's range it does not get transmitted. [22] aims at similar goals, using data caching and aggregation at each level of the tree.

In [23] it is introduced the direct diffusion of digests (aggregates), in which nodes compute their value as a function of their current value and the value received from neighbor nodes. And they start sending this new value, which can be piggy-backed in a periodic beacon. It takes at most the diameter of the network (in number of hops) to diffuse such digests. Obviously, in this form it only can compute digests of exemplary functions. Exemplary functions are the ones that can be computed as the result of an aggregate of previous values and one single new value. In fact, they have this name because their result depend on only one value, such as $\max$ and $\min$. For non-exemplary function, the authors propose that first, a direct digest that is always "won" by the sink is performed, second, each node would memorize the node from which it received the winning diffusion (parent) until a tree is formed, third, diffusion would be sent along the tree that emerged in the process. All the same, the authors do not show any difference between their proposal and the regular diffusion. Notwithstanding, the paper presents an interesting study of link asymmetry in WSN, which led the authors to propose a mechanism to switch parents whenever a certain link is asymmetric.

Nath *et al* [24] provide a formalization of the concept of diffusions. Three operations are considered synopsis generation, fusion and evaluation, that work as suggested by their names. The paper also provides a number of, so called, necessary and sufficient conditions for correctness, though, it also presents a situation that verifies all such conditions but does not produce the correct value, implying that the presented conditions are not sufficient. [25] uses a fuzzy logic approach to compute exemplary functions. Each node has a fuzzifier that decides whether to send the data. A comparison with the 'no aggregation scenario' was made, though no comparison with classical aggregations was presented. It was also used the sleep approach.

In [2] it is proposed to solve the problem of multi-path aggregation of duplicate sensitive nodes by keeping nodes from aggregating data if there is a possibility of duplication. To this end, upon the construction of the multi-path tree, nodes send theirs and their parents addresses along with their hop count to the root. Nodes that join the network, would know each of its parents and its parents' parents. Based on this information, if the node as more than one parent, then it chooses the parents parents with most paths from it as the aggregation point, otherwise it chooses its only parent as the aggregation point. Only two levels of the WSN are searched for, therefore it might happen that the link of the chosen parent's parents up the tree may fail while there is another parents' parents link which is operational. The fact that the lost of one (uptree) link/node can cause the loss of information of many of children nodes puts in question the very use of multi-path (redundancy). (The approach advocated in this paper has the property that for any set of link failures it can always compute the aggregate of the values that can still reach the sink.)

Manjhi *et al* [3] use an hybrid approach to routing. Using a tree approach closer to the leaves and a multi-path approach closer to the sink. This helps to leverage the advantages of trees (low latency, low messaging) with the advantages of multi-path (increased robustness).

Al-Karaki *et al* [26] present both an exact and an approximate algorithms to find optimal routes for aggregation in WNSs. The exact algorithm is stated as an integer linear programming problem, which the authors argue to be too complex to be solved in WSN. Hence, they propose an approximate genetic algorithm. However, the genetic algorithm is itself computationally heavy and involves many message exchanges, which is exacerbated by being performed in several rounds, thereby consuming the very same energy that the algorithm aims at saving.

Another type of aggregates that gain a certain moment in the community is the gossip based aggregation [27]–[32], in which each node 1) read its own value, 2)aggregates it with messages that it receives and 3) sends it to a random neighbor a fraction of the its current value while maintaining another fraction of it. After an algorithm-dependent number of rounds, the aggregates converge to the correct value. Usual objections include the apparent lack of benefits for the common WSN, higher latencies and increased number of communications.

Other types of optimizations to WSN that do not relate to the main contribution of this paper have been proposed, such as the exploitation of temporal correlation [33]–[38] and spatial correlation, such as [12], [14], [39]–[46].

A. Count Summary

The main contribution of this work is independent of which count summary is used. However, it requires that the count is performed prior to carrying out a duplicate-sensitive function, or at least, to have a cache with the nodes in the network. Therefore, it is paramount to use an efficient count summary that should be performed as infrequently as possible.

The following conditions reduce the need to perform count summaries: 1) parent nodes trigger a count summary action only if a given child does not communicate for a given period of time (T_{alive}), 2) all children send a message when they are initialized or whenever they notice that have a different parent node. Additionally, they also send at least one message (of whatever type) with a given time window (T_{alive}). The optimum value for T_{alive} is a compromise. On the one hand it must be as big as possible to not spoil too much energy. On the other hand, large T_{alive} impairs network reactivity.

To increase the efficiency of the count summary, nodes use a bitmap addressing, in which each node address corresponds to a bit in a address array. Addresses can be pre-programmed prior to the deployment of the network. Leaf nodes put themselves in the count summary by sending a message with their bit set. Aggregations of the count summary are performed by implementing a bitwise OR of partial results. The number of distinct nodes that are offspring of a given node is equal to the number of set bits on its bitmap array. Evidently, the number of nodes in the network is equal to the number of offspring of the root node plus one (the root itself). Due to the simple nature of this aggregated, a formal proof of its correctness will not be provided.

It should be remarked that a similar counting mechanism was proposed in [24]. However, their approach started with the bitmap addressing but at the higher levels used the FM-SKETCH [1], which conditions their approach to provide approximate results, and uses an high amount of memory [1].

III. Multi-path Aggregation

This paper considers mesh-like networks, where each node can reach only a limited subset of other network nodes, normally the nearest neighbors. Nodes' data should be forwarded to a particular node, designated by sink. Links that connect nodes are subject to errors, either transient or permanent. It is assumed that the underlying communication protocol provides error detection capabilities, discarding erroneous frames. Nodes are also subject to errors and are fail-silent, i.e., they either operate correctly or do not send any information. Furthermore, the following definitions apply:

- The WSN consists of $N_i, i = 0...K$ nodes.
- Without loss of generality, in the remainder of this paper N_0 designates the sink node;
- Each node is connected to one or more neighbour nodes;
- Each link between two nodes is designated by $L_{i,j}, i, j \in 0...K$;
- T_i is a bitmap representing the addresses of the node itself and its offspring
- A_k=[a,b,c, ...] denotes the aggregation of values a, b, c, ...;
- M_i designates a message sent by node N_i to its parent;
- A message can contain several aggregates. Aggregates sent in the same message are connected by a $+$ symbol.

The system undergoes three sequential phases: physical topology discovery, virtual topology set up and, finally, the data collection, which is the normal state. Permanent errors are considered as topology changes and thus this whole process may be repeated as often as necessary, although eventually only over specifics parts of the network affected by errors.

In the first phase all reachable nodes and its connections are identified. More concretely, each node discovers which nodes are its offspring. This is achieved, for example, by using a count summary as the one previously described in section II-A.

Once the topology is identified, phase 2, which consists in the virtual topology set-up, is started. At this stage each parent knows all the paths to each one of its offspring. Based on this information, each parent sends a message to each one of its children indicating if and how data should be aggregated, thereby creating several groups of aggregates. The decision about which data should be aggregated is taken primarily with a focus on minimizing the total number of messages.

Two different aggregation strategies are proposed in this paper, one more suitable to handle node errors and another more efficient in the presence of link errors. The difference between these strategies resides primarily in which messages are aggregated together. In the former case, messages are aggregated such that if all messages that come from a given child are lost the aggregates can still be *reconstructed*, provided that there is at least one redundant link, whereas in the latter case the focus is on messages lost on a given link. The best strategy to use in a particular WSN should be selected according with the most frequent error source.

Finally, after phase 2 is complete, the system enters the normal operation phase, in which the data is collected. During this phase each parent node receives messages from its offspring, eliminates or reconstructs data, depending on the existence of errors, and forwards the aggregates defined during phase 2 to its parent node. Data recovery requires only a few table look-ups and simple algebraic operations on the received aggregates, thus during normal operation the processing overhead is small.

A. *Multipath Aggregation of Duplicate Sensitive Summaries — Node Error Case*

The strategy proposed to deal with node failures consists in setting one of the children to aggregate its own data with the data of its offspring, while the other children send several aggregates. These aggregates consist in its own data aggregated with the data of its offspring, followed by aggregates that contain the data that do not intersect with previous siblings in the same level. To illustrate this strategy, consider the simple WSN depicted in figure 1.

As depicted in figure 1, node N_1 sends a message composed only by one field, which aggregates its own data and the data of its offspring (M_1 = [1,4,5]). Node N_2 sends one message with two fields, one aggregating its own data with its offspring (A_1=[2,5,6]) and another field that contains the data that does not intersect T_1 (i.e. $T_2 \setminus T_1$; A_2 = [2,6]). Thus, M_2={[2,5,6]+[2,6]}. A similar procedure is applied to node N_3. Table I presents the aggregates conveyed by messages M_1 to M_3.

In case there are no errors, the sink receives aggregates $A_i = [1, 4, 5], [2, 5, 6], [2, 6], [3, 6, 7], [3, 7], i \in 1..5$, respectively contained in messages M_1, M_2 and M_3. Note that in the third node there are some duplicate entries, that were removed. It can be trivially verified that the correct value can be recovered by taking the last field of each one of these messages. Therefore, this strategy meets the first goal addressed in this

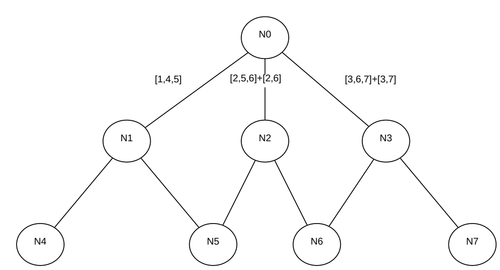

Fig. 1. Example 1: Node Error Strategy

[1]approximate queries are used primarily to reduce the amount of time and memory used to perform a given action, whereas exact queries are used when the correctness of the results have primacy

child index	message to send
1	$\{T_1\}$
2	$\{T_2\} + \{T_2 \backslash T_1\}$
3	$\{T_3\} + \{T_3 \backslash T_1 \cup T_3 \backslash T_2\} + \{T_3 \backslash \{T_3 \backslash T_1 \cup T_3 \backslash T_2\}\}$
$\vdots$	$\vdots$

TABLE I
EXAMPLE OF AGGREGATES FOR NODE ERRORS

paper, which is the ability to remove duplicates in the presence of redundant paths.

Lets now consider the case in which one of the nodes fails, e.g. node N_1. Observing figure 1, it can be seen that data from node N_1, which failed, and from node N_4, which has no redundant path to the sink, will be lost. However, data from all the other nodes should be recoverable. In fact, the sink node receives aggregates [2,5,6] + [2,6] and [3,6,7] + [3,7], respectively contained in messages M_2 and M_3. Data from all accessible nodes can be recovered by combining the first field of M_2 with the last field of M_3. If the failing node is node N_2, the sink node receives aggregates [1,4,5] and [3,6,7] + [3,7]. Data from all accessible nodes can be obtained by combining aggregates [1,4,5] and [3,6,7]. A similar reasoning could be carried out regarding the failure of any node in the network. Therefore, the proposed strategy meets the second goal of the paper, which is the ability to recover data that has redundant paths to the source in the presence of node errors.

This idea can be extended to a larger number of children, as shown in table I, i.e. each children first aggregate all its offspring, then aggregate all of its offspring minus the nodes that are reachable by nodes above it in the table, then do the same with minus sets from two nodes in above, then three and so on.

It can be seen that this approach may lead to an relatively high number of messages that must be sent by nodes further down the table, since the worst case number of aggregates grows as a power of two. Methods to dramatically reduce the number of entries of such table area addressed latter on.

B. Multipath Aggregation of Duplicate Sensitive Summaries — link Error Case

The link error case requires that whenever a link fails, the parent (or the root) node should receive all the data necessary to compute all aggregates, provided that there exists an alternative path.

The algorithm proposed to achieve this goal consists in having each node sending two aggregates, one containing the data related with its descendants that are reachable only by itself, and another aggregate composed by the data pertaining to nodes that communicate both with itself and its siblings. The algorithm resembles the one presented for the node error case, except that in the error node case the operation was a bitwise complement, whereas in this case (link error) the sets are disjoint in relation with the other children's children count summaries.

To illustrate this strategy, consider the simple WSN depicted in figure 2.

As depicted in figure 2, node N_1 sends a message composed by two fields, one aggregating its own data and the data of descendants that are reachable only by itself (node N_4, in the present case) and another aggregate with the data of descendants shared with each one of its siblings (only node N_5, in the present case), therefore M_1 = {[1,4]+[5]}. Node N_2 sends one message with three fields, one aggregating its own data only, since all its descendants are shared with its siblings, and two other aggregates with data of descendants shared with each one of its siblings, therefore M_2 = {[2]+[5]+[6]}. Following a similar reasoning, M_3 = {[3,7]+[6]}.

In this simple case it can be checked, by exhaustion, that this is a solution to the problem. In the absence of errors, the sink receives aggregates $A_i = [1, 4], [5], [2], [5], [6], [3, 7], [6], i \in 1..7$. The exact value can be recovered by adding A_1, A_2, A_3, A_5 and A_6. Thus, without errors the exact value can be recovered even in the presence of redundant paths, by sequentially adding the aggregates that have values not added before. By simple inspection of the aggregate set received by the sink, it can also be observed that each value appears in as many aggregates as the number of distinct paths to the sink. E.g. node N_5 that has two links appears in sets A_2 and A_4, while node N_4 appears only in one aggregate, since it has no redundant link. Thus, the redundancy is visible at the sink and it should be possible to recover the exact aggregate value even in the presence of errors. E.g. if link $L_{5,1}$ fails, the sink receives aggregates $A_i = [1, 4], [x], [2], [5], [6], [3, 7], [6]$. The exact value can be obtained e.g. by taking A_1, A_3, A_4, A_5 and A_6. The same reasoning can be applied to other sets to confirm that it is possible to recover the exact value of an aggregate function, even in the presence of link errors, by simple algebraic manipulation of the aggregates received by the sink, provided that there at least one alternative path.

C. Aggregate Generation and Data Reconstruction Algorithms

In previous sections, two algorithms to generate aggregates were introduced and illustrated in simple scenarios. Algorithms 1 and 2 describe how the aggregates can be generated for arbitrarily large WSN.

Even for the simple cases presented before, it was obvious that a rather high number of message exchanges could be necessary. In fact, in both cases the worst-case number of messages grew as a power of two with the number of siblings.

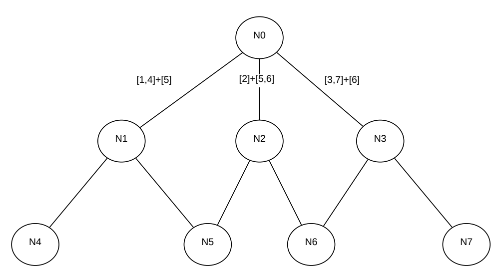

Fig. 2. Example 1: Link Error Strategy

Algorithm 1 Node Error Case: Parent Generated Transmission Lists

```
T_i ← Set of nodes reachable by offspring i
P← {} (Nodes that have been processed)
for {i = 1; i ≤ number of offsprings; i = i + 1} do
   M_i ← {}
   for {j = 0; j < i; j = j + 1} do
      Z ← Permutation of P taken j by j
      for all Z_l (elements of Z) do
         M_i ← M_i ⋃ {T_i \ ⋃_{x∈Z_l} T_x}
      end for
      j ← j + 1
   end for
   P← P ⋃{i}
   \\Remove duplicates and empty sets
   for j = 1; j < 2^{i−1} − 1; j = j + 1 do
      if M_i(j) = {} then
         remove M_i(j)
      else
         for k = j + 1; k < 2^{i−1}; k = k + 1 do
            if M_i(k) =M_i(j) then
               remove M_i(k)
            end if
         end for
      end if
   end for
end for
```

However, it should be noted that some of the entries of the aggregation tables are empty sets, while others are repetitions. An empty set occurs, for example, if all offsprings reached by a given node can also be reached by at least one of its siblings. And a repetition may happen if, for example, the set of offspring that a node can reach minus the set of nodes from a given sibling is equal to a similar set excluding another siblings offspring. Therefore, the algorithms herein presented also include an optimization section, in which the generated aggregate set is pruned of duplicate and empty sets, thus reducing the number and size of exchanged messages.

Eliminating redundancies (duplicates and empty sets) allows to perform a significant reduction of messages and aggregates. In sparse networks this optimization would not make much difference, since it is more likely that each node would have a rather distinct set of offspring[2]. However, as the network density grows, there would be more nodes with rather similar offspring tables, thereby the use of this improvement tend to become significant. Nonetheless, it must be stressed that node density control is out of the scope of this paper.

Algorithm 3 describes how the aggregates can be recovered from the received messages in the case of node errors. For each aggregate that a node manages (Q_k), it must inspect the aggregates sent by each of its children (R_i). Then, all the entries that failed are removed from u. Finally, the correct aggregate value is obtained by taking the rightmost element of u that has been received from each descendant. As can be verified, the algorithm is not computationally intensive. The operations carried out are relatively simple and the number of iterations depends on the number of aggregates and on the number of children, which are frequently relatively low values.

Algorithm 4 describes how the aggregates can be recovered

[2]this is not a problem since under this circumstances each node would send only one message with the aggregate of its own value and all its children, i.e. sparse networks have less paths in its multi-path

Algorithm 2 Link Error Case: Parent Generated Transmission Lists

```
T_i ← Set of nodes reachable by offspring i
N_i ← number of offsprings
for {i = 1; i ≤ N ; i = i + 1} do
   M_i ← {}
   Z ← Arrangements of {0, 1} with repetition taken N−1 by N−1
   for all Z_l (elements of Z) do
      M_i ← M_i ⋃ {T_i ⋂_{Z_l(x)=1} T_x ⋂_{Z_l(x)=0} T̄_x}
   end for
end for
\\Remove duplicates and empty sets
for j = 1; j < 2^{i−1} − 1; j = j + 1 do
   if M_i(j) = {} then
      remove M_i(j)
   else
      for k = j + 1; k < 2^{i−1}; k = k + 1 do
         if M_i(k) =M_i(j) then
            remove M_i(k)
         end if
      end for
   end if
end for
```

Algorithm 3 Node Error Case: Aggregate Computation From Children Messages

```
Q_k ← Set of aggregates that a node manages
R_i ← Set of aggregate values received from child i
RC_i ← Binary vector. 1 if received message form child i, 0 otherwise
U_k ← Set of reconstruction path to k^th aggregate
A← {} \\ A_k ← value of aggregate k
for {k = 1; k ≤ #[Q_k]; k = k + 1} do
   A_k ← self
   u← U_k
   for {i = 1; i ≤ #[R_i]; i = i + 1} do
      if not RC_i then
         for {j = i; j ≤ #[U_j]; j = j + 1} do
            remove instances of u related to node j
         end for
      end if
      A_k ← A_k+R_i(arg max u(i) = 1)
   end for
end for
```

Algorithm 4 Link Error Case: Aggregate Computation From Children Messages

Q$_k$ ← Set of aggregates that the node handles
R$_i$ ← Set of aggregate values received from child i
RC$_i$ ← Binary vector. 1 if received message form child i, 0 otherwise
U$_k$ ← Set of reconstruction path to k^{th} aggregate
A← {} \\ A$_k$ ← value of aggregate k
for $\{k = 1; k \leq \#[Q_k]; k = k + 1\}$ **do**
  A$_k$ ← self
  E← {} \\set of excluded aggregates
  for $\{i = 1; i \leq \#[R_i]; i = i + 1\}$ **do**
    u← $U_k \setminus U_k(RC_i = 1)$
    u← $u \setminus E$
    A$_k$ ←$aggr$\{A$_k$,R$_i$(u)\}
    E← $E \bigcup$\{u\}
  end for
end for

from the received messages in the case of link errors. The process consists in taking the aggregates sent by each children sequentially, and merge them if they have not been already included in a previous children. To keep track of which values have already been processed, it is used an exclusion list (E). In each step this list is appended with the index of all values that have been correctly received ($RC_i = 1$) and not yet merged.

In terms of computational complexity, the algorithm is similar to the previous one.

IV. Evaluation

This section presents simulation results to assess the effectiveness of the approaches proposed in this paper. The simulations were carried out in the Matlab® software, with a standard error model, in which the error probability is as function of the distance $P_r = P_e d^{-\alpha}$, with $\alpha = 2$, transmission power was equal in all the nodes, and the probability of error as $(1 - erf(P_r/N_o))/2$.

In addition to the two algorithms presented in this paper, this section also presents simulations of TAG and the DAG approaches, described in section II. The TAG approach is the simplest of all. It does not have any type of redundancy, using only simple aggregation, thus it will be used as the base line. The DAG approach has two level deep link error correction capability, hence it will be used compare with link error approach presented herein. To the best of the authors knowledge, there is no approach that can be used to make a direct comparison with the node error case.

The simulation was made with a network with 9 nodes placed randomly with a uniform distribution in a square of side $2.7m$. The sink was chosen randomly, therefore, not being necessarily in one of the edges of the network. Communication parameters were tuned to ensure a communication range of about $2m$ with an error probability of 0.1. 100 simulations were done on the same network for each case, i.e. each protocol and each error mode. All nodes were programmed to act as if they had read a 1 from their respective sensor (including the sink) which allowed to have a global view into the number of nodes that were successfully aggregated. Recall that TAG is single path, hence in all case the nodes that were aggregated, were done so once. Results are as follows:

From figure 3, the link error case behaved as expected with approach tuned to link case having most of its occurrences with the correct reception of all nodes. The DAG approach also presented a reasonable/similar behavior. From figure 4, the node error case presents a small discrepancy with the expectation, i.e. the DAG approach behaved a little bit better than our approach tuned to the error case, there are a few reasons for this, 1)the DAG approach has a two level deep link error correction capabilities and since the network was small it could correct most of the errors (in fact, DAG in two

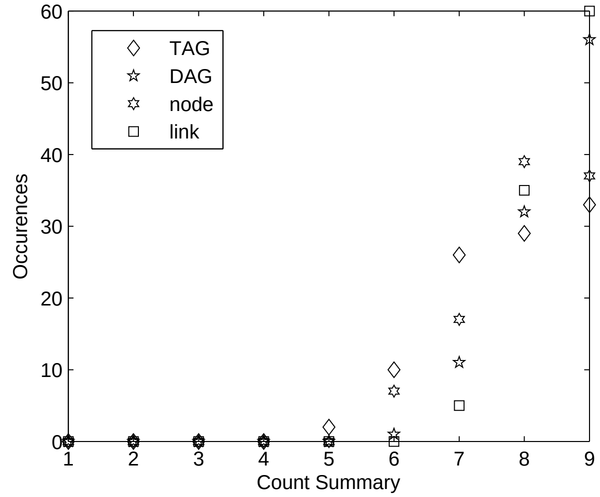

Fig. 3. Link Error Case.

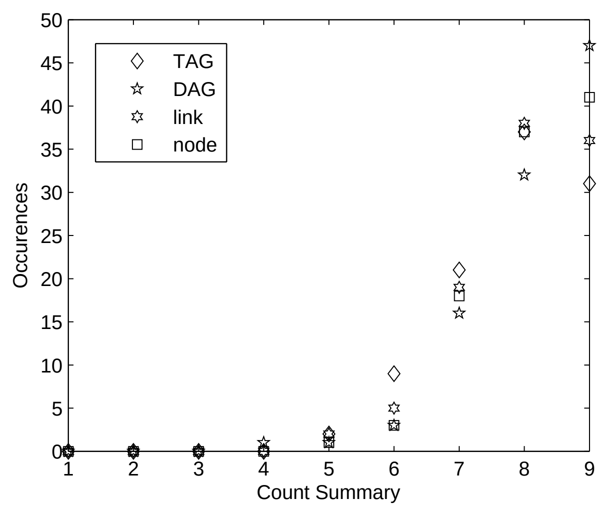

Fig. 4. Node Error Case.

level, i.e. sink plus two, is similar to our link scheme) and 2)the experiences were done in a single network which potentially means that a given protocol could have, out of serendipity, have ended up in a network with many nodes in level 1. These results call for simulations with more nodes, more levels and also simulations with several different networks.

V. Conclusions

WSN present data transmission/reception reliability issues, which can be dealt with by the use of multi-path routing. Nonetheless, this solution introduces another problem, namely aggregate reliability in the presence of duplicate-sensitive summaries. This paper presented a mechanism that ensures that the value of such aggregates will be as close as possible to the actual value, namely by taking advantage of redundant paths in the presence of errors and removing duplicates.

The mechanism is focused in the partition of the summaries into several messages that are recomposed to form the best possible message in their way to the sink. Two of such algorithms were devised, one best suited for networks in which the dominant failure mode is link failure and another in which the dominant failure mode is node failure.

Both scenarios were simulated and compared to the standard approach in the literature, having demonstrated a superior performance in their respective failure mode scenarios. However, there was also a reduction in the lifetime of the network.

Future work consists in providing formal proofs of the correctness of the proposed algorithms to generate the aggregates and recover the data, as well as consider less pessimistic failure modes, namely by controlling the number of nodes or links that may fail simultaneously.

References

[1] J. Considine, F. Li, G. Kollios, and J. Byers, "Approximate aggregation techniques for sensor databases," in *Data Engineering, 2004. Proceedings. 20th International Conference on*, march-2 april 2004, pp. 449 – 460.

[2] S. Motegi, K. Yoshihara, and H. Horiuchi, "Dag based in-network aggregation for sensor network monitoring," in *Applications and the Internet, 2006. SAINT 2006. International Symposium on*, jan. 2006, pp. 8 pp. –299.

[3] A. Manjhi, S. Nath, and P. B. Gibbons, "Tributaries and deltas: efficient and robust aggregation in sensor network streams," in *Proceedings of the 2005 ACM SIGMOD international conference on Management of data*, ser. SIGMOD '05. New York, NY, USA: ACM, 2005, pp. 287–298. [Online]. Available: http://doi.acm.org/10.1145/1066157.1066191

[4] D. Wu and M. H. Wong, "Fast and simultaneous data aggregation over multiple regions in wireless sensor networks," *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, vol. 41, no. 3, pp. 333 –343, may 2011.

[5] S. K. Chong, M. M. Gaber, S. Krishnaswamy, and S. W. Loke, "Energy-aware data processing techniques for wireless sensor networks: A review." *T. Large - Scale Data - and Knowledge-Centered Systems*, vol. 3, pp. 117–137, 2011.

[6] B. Krishnamachari, D. Estrin, and S. B. Wicker, "The impact of data aggregation in wireless sensor networks," in *Proceedings of the 22nd International Conference on Distributed Computing Systems*, ser. ICDCSW '02. Washington, DC, USA: IEEE Computer Society, 2002, pp. 575–578. [Online]. Available: http://dl.acm.org/citation.cfm?id=646854.708078

[7] K. Kalpakis, K. Dasgupta, and P. Namjoshi, "Maximum lifetime data gathering and aggregation in wireless sensor networks," 2002.

[8] A. Woo, S. Madden, and R. Govindan, "Networking support for query processing in sensor networks," *Commun. ACM*, vol. 47, pp. 47–52, June 2004. [Online]. Available: http://doi.acm.org/10.1145/990680.990706

[9] M. Watfa, W. Daher, and H. Al Azar, "A sensor network data aggregation technique," *International Journal of Computer Theory and Engineering*, vol. 1, no. 1, pp. 19–26, April 2009.

[10] S. Madden, M. J. Franklin, J. M. Hellerstein, and W. Hong, "Tag: a tiny aggregation service for ad-hoc sensor networks," *SIGOPS Oper. Syst. Rev.*, vol. 36, pp. 131–146, December 2002. [Online]. Available: http://doi.acm.org/10.1145/844128.844142

[11] W.-H. Liao, Y. Kao, and C.-M. Fan, "An ant colony algorithm for data aggregation in wireless sensor networks," in *Proceedings of the 2007 International Conference on Sensor Technologies and Applications*, ser. SENSORCOMM '07. Washington, DC, USA: IEEE Computer Society, 2007, pp. 101–106. [Online]. Available: http://dx.doi.org/10.1109/SENSORCOMM.2007.23

[12] S. Okdem and D. Karaboga, "Routing in wireless sensor networks using an ant colony optimization (aco) router chip," *Sensors*, vol. 9, no. 2, pp. 909–921, 2009. [Online]. Available: http://www.mdpi.com/1424-8220/9/2/909/

[13] D. Ganesan, B. Greenstein, D. Perelyubskiy, D. Estrin, and J. Heidemann, "An evaluation of multi-resolution storage for sensor networks," in *Proceedings of the 1st international conference on Embedded networked sensor systems*, ser. SenSys '03. New York, NY, USA: ACM, 2003, pp. 89–102. [Online]. Available: http://doi.acm.org/10.1145/958491.958502

[14] J. Gao, L. Guibas, N. Milosavljevic, and J. Hershberger, "Sparse data aggregation in sensor networks," in *Proceedings of the 6th international conference on Information processing in sensor networks*, ser. IPSN '07. New York, NY, USA: ACM, 2007, pp. 430–439. [Online]. Available: http://doi.acm.org/10.1145/1236360.1236414

[15] G. Cormode, M. Garofalakis, S. Muthukrishnan, and R. Rastogi, "Holistic aggregates in a networked world: distributed tracking of approximate quantiles," in *Proceedings of the 2005 ACM SIGMOD international conference on Management of data*, ser. SIGMOD '05. New York, NY, USA: ACM, 2005, pp. 25–36. [Online]. Available: http://doi.acm.org/10.1145/1066157.1066161

[16] N. Shrivastava, C. Buragohain, D. Agrawal, and S. Suri, "Medians and beyond: new aggregation techniques for sensor networks," in *Proceedings of the 2nd international conference on Embedded networked sensor systems*, ser. SenSys '04. New York, NY, USA: ACM, 2004, pp. 239–249. [Online]. Available: http://doi.acm.org/10.1145/1031495.1031524

[17] M. Wu, J. Xu, X. Tang, and W.-C. Lee, "Monitoring top-k query in-wireless sensor networks," *Data Engineering, International Conference on*, pp. 143–147, 2006.

[18] ——, "Top-k monitoring in wireless sensor networks," *IEEE Trans. on Knowl. and Data Eng.*, vol. 19, pp. 962–976, July 2007. [Online]. Available: http://dx.doi.org/10.1109/TKDE.2007.1038

[19] K. Ammar and M. A. Nascimento, "On the use of histograms for processing exact aggregate queries in wireless sensor networks," *International Workshop on Data Management for Sensor Networks*, vol. 8, 2011.

[20] A. Manjeshwar and D. Agrawal, "Teen: a routing protocol for enhanced efficiency in wireless sensor networks," in *Parallel and Distributed Processing Symposium., Proceedings 15th International*, apr 2001, pp. 2009 –2015.

[21] A. Manjeshwar and D. P. Agrawal, "Apteen: A hybrid protocol for efficient routing and comprehensive information retrieval in wireless sensor networks," in *Proceedings of the 16th International Parallel and Distributed Processing Symposium*, ser. IPDPS '02. Washington, DC, USA: IEEE Computer Society, 2002, pp. 48–55. [Online]. Available: http://dl.acm.org/citation.cfm?id=645610.662036

[22] M. A. Sharaf, J. Beaver, A. Labrinidis, and P. K. Chrysanthis, "Tina: a scheme for temporal coherency-aware in-network aggregation," in *Proceedings of the 3rd ACM international workshop on Data engineering for wireless and mobile access*, ser. MobiDe '03. New York, NY, USA: ACM, 2003, pp. 69–76. [Online]. Available: http://doi.acm.org/10.1145/940923.940937

[23] J. Zhao, R. Govindan, and D. Estrin, "Computing aggregates for monitoring wireless sensor networks," in *Sensor Network Protocols and Applications, 2003. Proceedings of the First IEEE. 2003 IEEE International Workshop on*, may 2003, pp. 139 – 148.

[24] S. Nath, P. B. Gibbons, S. Seshan, and Z. Anderson, "Synopsis diffusion for robust aggregation in sensor networks," *ACM Trans. Sen. Netw.*, vol. 4, pp. 7:1–7:40, April 2008. [Online]. Available: http://doi.acm.org/10.1145/1340771.1340773

[25] S. Croce, F. Marcelloni, and M. Vecchio, "Reducing power consumption in wireless sensor networks using a novel approach to data aggregation," *Comput. J.*, vol. 51, pp. 227–239, March 2008. [Online]. Available: http://dl.acm.org/citation.cfm?id=1349558.1349559

[26] J. Al-Karaki, R. Ul-Mustafa, and A. Kamal, "Data aggregation in wireless sensor networks - exact and approximate algorithms," in *High Performance Switching and Routing, 2004. HPSR. 2004 Workshop on*, 2004, pp. 241 – 245.

[27] A. Dimakis, A. Sarwate, and M. Wainwright, "Geographic gossip: efficient aggregation for sensor networks," in *Information Processing in Sensor Networks, 2006. IPSN 2006. The Fifth International Conference on*, 2006, pp. 69–76.

[28] D. Kempe, A. Dobra, and J. Gehrke, "Gossip-based computation of aggregate information," in *Foundations of Computer Science, 2003. Proceedings. 44th Annual IEEE Symposium on*, oct. 2003, pp. 482 – 491.

[29] R. Sarkar, X. Zhu, and J. Gao, "Hierarchical spatial gossip for multi-resolution representations in sensor networks," in *Information Processing in Sensor Networks, 2007. IPSN 2007. 6th International Symposium on*, april 2007, pp. 420–429.

[30] K.-W. Fan, S. Liu, and P. Sinha, "On the potential of structure-free data aggregation in sensor networks," in *INFOCOM 2006. 25th IEEE International Conference on Computer Communications. Proceedings*, april 2006, pp. 1–12.

[31] S. Boyd, A. Ghosh, B. Prabhakar, and D. Shah, "Randomized gossip algorithms," *Information Theory, IEEE Transactions on*, vol. 52, no. 6, pp. 2508 – 2530, june 2006.

[32] J.-Y. Chen, G. Pandurangan, and D. Xu, "Robust computation of aggregates in wireless sensor networks: distributed randomized algorithms and analysis," in *Proceedings of the 4th international symposium on Information processing in sensor networks*, ser. IPSN '05. Piscataway, NJ, USA: IEEE Press, 2005. [Online]. Available: http://dl.acm.org/citation.cfm?id=1147685.1147741

[33] C. Olston, J. Jiang, and J. Widom, "Adaptive filters for continuous queries over distributed data streams," in *Proceedings of the 2003 ACM SIGMOD international conference on Management of data*, ser. SIGMOD '03. New York, NY, USA: ACM, 2003, pp. 563–574. [Online]. Available: http://doi.acm.org/10.1145/872757.872825

[34] C. Olston, B. T. Loo, and J. Widom, "Adaptive precision setting for cached approximate values," in *Proceedings of the 2001 ACM SIGMOD international conference on Management of data*, ser. SIGMOD '01. New York, NY, USA: ACM, 2001, pp. 355–366. [Online]. Available: http://doi.acm.org/10.1145/375663.375710

[35] C. Olston and J. Widom, "Best-effort cache synchronization with source cooperation," in *Proceedings of the 2002 ACM SIGMOD international conference on Management of data*, ser. SIGMOD '02. New York, NY, USA: ACM, 2002, pp. 73–84. [Online]. Available: http://doi.acm.org/10.1145/564691.564701

[36] A. Deligiannakis, Y. Kotidis, and N. Roussopoulos, "Hierarchical in-network data aggregation with quality guarantees." in *EDBT*, ser. Lecture Notes in Computer Science, E. Bertino, S. Christodoulakis, D. Plexousakis, V. Christophides, M. Koubarakis, K. Bhm, and E. Ferrari, Eds., vol. 2992. Springer, 2004, pp. 658–675.

[37] S. Shah, A. Bernard, V. Sharma, K. Ramamirtham, and P. Shenoy, "Maintaining temporal coherency of cooperating dynamic data repositories," in *In Proc. VLDB*, 2002.

[38] C. Olston and J. Widom, "Offering a precision-performance tradeoff for aggregation queries over replicated data," in *Proceedings of the 26th International Conference on Very Large Data Bases*, ser. VLDB '00. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2000, pp. 144–155. [Online]. Available: http://dl.acm.org/citation.cfm?id=645926.671877

[39] L. Rossi, B. Krishnamachari, and C.-C. Kuo, "Distributed parameter estimation for monitoring diffusion phenomena using physical models," in *Sensor and Ad Hoc Communications and Networks, 2004. IEEE SECON 2004. 2004 First Annual IEEE Communications Society Conference on*, oct. 2004, pp. 460–469.

[40] C. Guestrin, P. Bodik, R. Thibaux, M. Paskin, and S. Madden, "Distributed regression: an efficient framework for modeling sensor network data," in *Proceedings of the 3rd international symposium on Information processing in sensor networks*, ser. IPSN '04. New York, NY, USA: ACM, 2004, pp. 1–10. [Online]. Available: http://doi.acm.org/10.1145/984622.984624

[41] H. Gupta, V. Navda, S. R. Das, and V. Chowdhary, "Efficient gathering of correlated data in sensor networks," in *Proceedings of the 6th ACM international symposium on Mobile ad hoc networking and computing*, ser. MobiHoc '05. New York, NY, USA: ACM, 2005, pp. 402–413. [Online]. Available: http://doi.acm.org/10.1145/1062689.1062739

[42] W. R. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "Energy-efficient communication protocol for wireless microsensor networks," in *Proceedings of the 33rd Hawaii International Conference on System Sciences-Volume 8*, ser. HICSS '00, vol. 8. Washington, DC, USA: IEEE Computer Society, 2000, pp. 8020–. [Online]. Available: http://dl.acm.org/citation.cfm?id=820264.820485

[43] A. Deshpande, C. Guestrin, S. R. Madden, J. M. Hellerstein, and W. Hong, "Model-driven data acquisition in sensor networks," in *Proceedings of the Thirtieth international conference on Very large data bases - Volume 30*, ser. VLDB '04. VLDB Endowment, 2004, pp. 588–599. [Online]. Available: http://dl.acm.org/citation.cfm?id=1316689.1316741

[44] A. Jindal and K. Psounis, "Modeling spatially-correlated sensor network data," in *Sensor and Ad Hoc Communications and Networks, 2004. IEEE SECON 2004. 2004 First Annual IEEE Communications Society Conference on*, oct. 2004, pp. 162 – 171.

[45] M. C. Vuran and I. F. Akyildiz, "Spatial correlation-based collaborative medium access control in wireless sensor networks," *IEEE/ACM Trans. Netw.*, vol. 14, pp. 316–329, April 2006. [Online]. Available: http://dx.doi.org/10.1109/TENT.2006.872544

[46] S. Yoon and C. Shahabi, "The clustered aggregation (cag) technique leveraging spatial and temporal correlations in wireless sensor networks," in *ACM Trans. Sen. Netw.*, vol. 3. New York, NY, USA: ACM, March 2007. [Online]. Available: http://doi.acm.org/10.1145/1210669.1210672