

Impact on Accuracy of Deployment Trade-Offs in Localized Sensor Network Event Detection

Farhana Jabeen, Alvaro A. A. Fernandes
School of Computer Science
University of Manchester
Manchester M13 9PL, UK
Email: {f.jabeen,a.fernandes}@cs.man.ac.uk

Abstract—Event detection in sensor networks is a very desirable capability, insofar as it can be seen as a building block for higher-level functionalities. Of particular interest in this paper is the ability to characterize the geometry of physical phenomena (e.g., a weather front, or a forest fire) as they take place in the sensing scope of a deployed network. If such geometries can be accurately characterized in a cost-effective manner, then we can construe them as induced geometrical extents. Taken together with asserted geometrical extents (e.g., those of civil structures such as buildings and roads and bridges, or of geographical features, such as forests, rivers, lakes, mountains), physical phenomena can then be targetted by high-level, declarative spatial and spatio-temporal queries. For this vision to be realized, one must understand better the conditions under which best-of-breed event detection algorithms for sensor networks perform well on realistic geometries given realistic assumptions about deployment circumstances and operating conditions. In this paper, we contribute a study of two event detection algorithms, viz., FEBD and T-Fit. The study revealed shortcomings in both algorithms, which we used to develop improved versions of both. We have quantified the trade-offs between accuracy, areal coverage and sensor density that they incur. Our overarching goal has, therefore, been a practical one, viz., to inform future deployments as to what accuracy can be expected, given a desired areal coverage and such sensor density as can be afforded, by the use of the algorithms under evaluation.

I. INTRODUCTION

There is widespread agreement that wireless sensor networks are on their way to becoming an essential technology for monitoring the natural environment and for modelling the behaviour of physical phenomena over space and time [1]. This is testified by many reported deployments, e.g., close to fifty environmental sensor networks were surveyed in [2].

As construed by many, sensor networks (or, as some prefer, sensor webs [3], [4]) are macroscopes [5] (also referred to as macro-instruments [3]). Thus, “the dense temporal and spatial monitoring of large volumes that they provide offers a way to perceive complex interactions” [5]. They allow “for the spatio-temporal understanding of an environment through coordinated efforts” [3].

It is an integral part of this overarching vision that nodes in a sensor network cooperate in the execution of high-level tasks. Thus, [3] envisions that the sensor network is, essentially, a distributed computing device, i.e., “a parallel-type architecture as sensor measurements (including [sensor] location) are passed, and collectively interpreted, from [sensor] to [sensor]”.

In this context, this paper focus on the distributed computation of a particular interpretation task, viz., the characterization of the geometry of physical phenomena from sensing data. In the literature, this has been referred to as the problem of sensor network event (or edge, or boundary) detection (see, e.g., [6], [7], [8]). We will refer to it, acronymically, as *SNED*, or as *the SNED problem*.

The remainder of this paper is structured as follows. In Section II, we define the *SNED* problem [6], and describe two localized algorithms, viz., FEBD [7] and T-Fit [8], that are amongst the best current solutions to *SNED* reported in the literature as well as a modified version of each that we ourselves developed in response to shortcomings of the original algorithms that our experimental work has revealed. Section III is devoted to a study of how, in the case of example events derived from natural phenomena documented in the literature [9], [10], the performance of FEBD and T-Fit varies as deployment-level parameters (such as areal coverage and sensing density) vary. We motivate our study, describe the example events we have used, explain our experimental design, report the experimental results we have obtained, and, finally, analyze and discuss them. In Section IV, we draw some conclusions.

II. EVENT DETECTION IN SENSOR NETWORK

A. Problem Definition

Sensor nodes can be used to detect phenomena that are extended in space (e.g., contaminant flows, or microorganism concentrations, etc.). We specifically consider those that can be modelled as having a boundary, or edge. Examples include pressure contours, temperature gradients, variations in levels of measurable quantities such as light intensity, chemical concentration, etc..

Consider a network that is sensing a field. Let the field be decomposable into two or more regions of distinct behaviour (e.g., different mean values for the sensor measurements of a given quantity). In this case, there exist boundaries that are diagnostic of region membership (e.g., regions which fall into distinct temperature ranges). Intuitively, the *SNED* problem can be formulated as follows: given a set of readings from sensor located at different points in the field and an event characterization predicate (e.g., a membership test for

temperature range), return a description of the boundary of the event (i.e., the points in the field satisfying the event predicate).

The assumptions that are typically made are that the network uses wireless communication, nodes are tethered (or at least have their location known) and energy-bounded, readings are synchronized but inaccurate with some probability; that the event characterization predicate takes the form of threshold test, possibly over the discretization of a continuous value range onto a number of bands (or bins); and that the returned description is a representation of the (possibly closed) polyline which is (or encompasses) the event.

Consider Fig. 1. It depicts a sensor network deployed as regular grid in which there is a sensor in each point of the field. The left figure depicts, by the presence or absence of shading, two regions of distinct behaviour. The right part of Fig. 1 exemplifies what the outcome of event detection might be. The darkest points mark the ideal sensed boundary of the event, the darker-shaded points inside the polygon have also detected the events but are not in its boundary. Note that, as expected, the sensing density is one factor determining the representational fidelity. In particular, note that the actual sensed boundary may have a non-negligible thickness, indicated in the right part of Fig. 1 by a thick, lighter-shaded line, which reports more nodes, in a lighter shade (viz., (1,3), (2,3), (2,4), (2,5), (6,2), (6,5), (6,6) and (7,2)), as boundary nodes than would otherwise be the case.

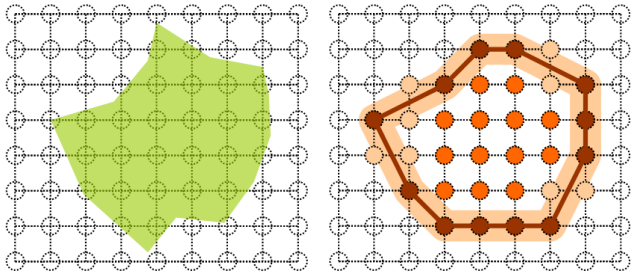


Fig. 1. Actual Event Geometry (left); Sensed Event Geometry (right)

The **SNED** problem is quite challenging, even though the formulation of above could be made more challenging still (e.g., by taking nodes to be mobile). The main reason is that any practical solution must be a localized algorithm for in-network processing. This requirement arises because, as is well known, the cost of communication dominates energy consumption and nodes are energy-bounded. The need to reduce communication precludes sending all readings back to base as well as any scheme that requires exchange of messages beyond the neighbourhood of a node. Thus, the general form of solutions to **SNED** that are of interest in this paper requires each node to locally determine whether it lies in or near the boundary of the given event. In order to do so, the node may receive (and send) information to its k -hop neighbours.

B. Proposed Solutions

There have been many proposed solutions for the **SNED** problem. We now briefly survey the most prominent ones.

To the best of our knowledge, the first characterization of **SNED** was reported in [6]. In that paper, the authors propose three different approaches to **SNED**: statistical, image-processing, and classifier-based. A statistical approach requires a node to collect information from its neighbours, derive a set of statistics from that information and use a Boolean decision function in order to decide whether it lies in the event boundary (often, but not always) based on an acceptance threshold. The image-processing approach treats each sensor as a pixel, thereby opening the way for the direct application of edge-detection techniques used in the treatment of images, e.g., those based on computing a filtered image using convolution. Note that if node deployment is such that the required pixel-like regularity is not observed, a weighting scheme can be used. Finally, in the classifier-based approach, a node attempts to partition the information collected from its neighbourhood into regions of distinct behaviour using classification techniques. The authors then instantiate each approach and comparatively evaluate their instantiations using detection accuracy and boundary thickness as their metrics. For those particular metrics and instantiations, the classifier-based approach both produced better results and was shown to be susceptible to degradation in performance in the presence of sensor errors.

Since sensor errors are assumed likely, many researchers have built upon [6] and focussed on this aspect of the solution as their priority [11], [12], [13], [7], [14]. However, the issue of robustness in the presence of sensor errors can be seen as orthogonal, a position implicitly held in [6], as well as by other recent proposals (e.g., [8]).

In this paper, we perform a comparative evaluation of two solutions to the **SNED** problem, viz., FEBD [7] and T-Fit [8]. These two algorithms were chosen because they were shown in [7] and [8] to be competitive with respect to the then state-of-the-art and are both simple to understand and implement (and hence also simple to evaluate and analyze). In addition, they are still conceptually distinct enough to suggest that the quality of their outcomes might not be strictly positively correlated.

Both [7] and [8] report, as expected, the outcome of *analytical* studies, i.e., experiments that ultimately describe the behaviour of the reported solution given certain inputs. However, and again as expected, neither [7] nor [8] is a *comparative* study, i.e., neither of them attempts to determine experimentally, for the same inputs, how the reported solution compares vis-à-vis any other. Moreover, the experiments in one paper are, to a great degree, incomparable with respect to those in other papers.

There is a significant practical drawback in this state of affairs: if an environmental scientist were to try and seek guidance in the literature as to which, among proposed **SNED** solutions, would perform better for a planned deployment, she would find the absence of comparative evaluations a serious impediment for an informed decision.

Therefore, the remainder of this paper focusses on describing FEBD and T-Fit and comparatively evaluating them with

inputs that are realistic (i.e., consist of events, adapted from those reported in [10], [9], whose geometry occurs in nature, rather than the Platonic forms (such as circles, squares and ellipses) used in the original papers.

C. The FEBD Algorithm

Fig. 2 is a precise specification of FEBD derived from the informal account given in [7].

FEBD imposes an algorithmic structure onto the solution that can be seen to comprise the following phases: (i) sensing (l. 3), (ii) communication (ll. 6–10), (iii) error-suppression (l. 11), and (iv) edge detection, properly said (ll. 12–14).

For the purposes of later contrast, we observe the following facts about FEBD. (A) FEBD requires every node to send the outcome of its event detection predicate to all its one-hop neighbours, irrespective of whether that outcome was true or false. (B) The scheme used by FEBD to contend with sensor errors is to subject a node’s own call as to whether it is an event node to a majority rule. Thus, the call of every one-hop neighbour of a node is counted. If the majority vote concurs with the node’s call, then that call is allowed to stand, otherwise not. (C) FEBD takes as input an acceptance threshold (see [7] for a specific suggestion as to how to compute it). This is then used to determine whether an event node is an edge node (by the statistic computed in l. 7 of the FEBDEDGEDETECTION algorithm in Fig. 2).

D. CFEBD: A Cautious Version of FEBD

Our empirical study (reported in Sec. III) revealed shortcomings in the original scheme used by FEBD for contending with sensing errors. As presented in Fig. 2 (and in [7]), the fact that a node has detected an event must be confirmed by a majority vote where the pool of voters consists of all nodes in the neighbourhood defined by the radio range RR . We have identified in our experiments that for geometries in which (roughly speaking) narrow bands or acute jagged edges occur, the scheme in [7] produces poor outcomes if the node density is high (equivalently, in the case of a regular grid deployment, if the distance d between nodes is small). Essentially, in this case, FEBD is overconfident in judging a measurement to be erroneous because it gives voting rights to too many nodes. The refinement we propose is that when $d < RR/2$, i.e., when the density is considered to be high, the pool of votes is shrunk to comprise only those nodes in the half-neighbourhood of the node, i.e., the neighbourhood defined by the radius $RR/2$. Because this approach makes the original FEBD more cautious when judging a measurement to be erroneous, we call this modified version *cautious* FEBD or, CFEBD, for short.

E. The T-Fit Algorithm

Fig. 3 is a precise specification of T-Fit derived from the informal account given in [8].

We observe that T-Fit follows the same algorithmic structure as FEBD. By way of contrast, we note the following facts about T-Fit. (A) Unlike FEBD it only requires a node to send

the outcome of its event detection predicate to all its one-hop neighbours, if that outcome is positive, i.e., if the node makes a call that it is an event node. The implications of this for the comparative message complexity of the two algorithms are, of course, significant. (B) T-Fit has no error-suppression, i.e., phase (iii) of the solution can be said to be a no-op. The implications of this for the comparative accuracy of the two algorithms are, of course, significant. (C) T-Fit does not require an acceptance threshold in its edge detection phase. This is significant in that it makes the algorithm comparatively easier to deploy in practice as it eliminates the need for a judicious choice of a parameter, which is always a difficult, error-prone task for a deployer to perform. (D) T-Fit uses geometrical notions to call an event node an edge node. At each node, it partitions the node’s one-hop neighbourhood into quadrants defined by a circle centred at the node with the radio range RR as radius. The edge-detection statistic is then formulated in terms of the population of the quadrants and of the angles formed with the centre node by its most distant neighbours in diagonal quadrants (see Fig. 3 and [8]).

F. CFT-Fit: A Cautious, Fault-Tolerant Version of T-Fit

Our empirical study (reported in Sec. III) revealed shortcomings in T-Fit which led us to modify the original algorithm in three principal ways, as follows. As presented in [8], the algorithm has provision for contending with sensor error. As exemplified by FEBD (see Fig. 2), it is possible for a solution to **SNED** to treat the problem of sensor errors in a well-defined algorithmic step. We have, therefore, added to the original T-Fit algorithm an error-suppression phase, and, for comparability, we used the error suppression scheme used in CFEBD above. This modification makes the algorithm more realistic but has a the side-effect of increasing the message complexity of the original algorithm to that of FEBD (and CFEBD).

We have also identified in our experiments that when the event is not connected (i.e., it exhibits the geometry of an archipelago, in pictorial terms), then the assumption that a node whose four quadrants contain event nodes is, for that reason alone, not an edge node (see Fig. 3, bottom, l. 20) is too optimistic. We have introduced a modification that has the effect of introducing more caution into this decision, as follows. When an event node has neighbours that are event nodes in all four quadrants but in any one of those quadrants, at least half of the nodes are not event nodes, then that quadrant is considered abnormal and we apply the more precise rule for the case in which there are nodes in three quadrants ((see Fig. 3, bottom, ll. 23–26)), in this case, the three remaining quadrants.

Finally, in the original algorithm, there is some ambiguity as to what procedure one should follow when *all* the neighbouring nodes lie in the vertical or horizontal axes with respect to the centre node. We have, therefore, chosen to treat such cases explicitly, as follows. If there are neighbours in all four axes, then an event node is not an edge node. If there are neighbours only in one, or in two, or in three axes, then an event node is

```

FEBD( $SN :: \text{list}(\text{sensorId})$ ,  $\text{eventPredicate} :: \text{float} \rightarrow \text{Boolean}$ ,  $\text{acceptThreshold} :: \text{float}$ )
1  for  $s \in SN$ 
2    do in parallel over [ $s$ ]
3       $\text{sensorReading} :: \text{float} \leftarrow \text{ACQUIREREADING}()$ 
4       $\text{eventDetected} :: \text{Boolean} \leftarrow \text{eventPredicate}(\text{sensorReading})$ 
5       $\text{outgoingMsg} :: \text{pair}(\text{sensorId}, \text{Boolean}) \leftarrow (s, \text{eventDetected})$ 
6      for  $n \in \text{ONEHOPNEIGHBOURHOOD}(s, SN)$ 
7        do [ $n$ ]  $\text{SEND}(\text{outgoingMsg})$ 
8       $\text{incomingMsgs} :: \text{list}(\text{pair}(\text{sensorId}, \text{Boolean})) \leftarrow []$ 
9      for  $n \in \text{ONEHOPNEIGHBOURHOOD}(s, SN)$ 
10       do [ $n$ ]  $\text{incomingMsgs}.\text{APPEND}(\text{RECEIVE}(n))$ 
11      $\text{eventConfirmed} :: \text{Boolean} \leftarrow \text{FAULTSUPPRESSION}(\text{incomingMsgs}, \text{eventDetected})$ 
12     if  $\text{eventConfirmed}$ 
13       then  $\text{edgeDetected} :: \text{Boolean} \leftarrow \text{FEBDEDGEDETECTION}(\text{incomingMsgs}, \text{acceptThreshold})$ 
14       else  $\text{edgeDetected} :: \text{Boolean} \leftarrow \text{false}$ 
15     return  $(s, \text{edgeDetected})$ 

FAULTSUPPRESSION( $\text{incomingMsgs} :: \text{list}(\text{pair}(\text{sensorId}, \text{Boolean}))$ ,  $\text{eventDetected} :: \text{Boolean}$ )
1   $\text{neighbourhoodSize} :: \text{int} \leftarrow \text{LENGTH}(\text{incomingMsgs})$ 
2   $\text{neighbourAgreements} :: \text{int} \leftarrow 0$ 
3  for  $m \in 1$  to  $\text{neighbourhoodSize}$ 
4    do if  $\text{incomingMsgs}[m].\text{eventDetected} = \text{eventDetected}$ 
5      then  $\text{neighbourAgreements} \leftarrow \text{neighbourAgreements} + 1$ 
6  return  $\text{neighbourAgreements} \geq (\text{neighbourhoodSize} - \text{neighbourAgreements})$ 

FEBDEDGEDETECTION( $\text{incomingMsgs} :: \text{list}(\text{pair}(\text{sensorId}, \text{Boolean}))$ ,  $\text{acceptThreshold} :: \text{float}$ )
1   $\text{neighbourhoodSize} :: \text{int} \leftarrow \text{LENGTH}(\text{incomingMsgs})$ 
2   $\text{neighbouringDetections} :: \text{int} \leftarrow 0$ 
3  for  $m \in 1$  to  $\text{neighbourhoodSize}$ 
4    do if  $\text{incomingMsgs}[m].\text{eventDetected}$ 
5      then  $\text{neighbouringDetections} \leftarrow \text{neighbouringDetections} + 1$ 
6   $\text{neighbouringNonDetections} :: \text{int} \leftarrow (\text{neighbourhoodSize} - \text{neighbouringDetections})$ 
7   $\text{edgeDetected} :: \text{Boolean} \leftarrow ((1 - ((\text{neighbouringDetections} - \text{neighbouringNonDetections}) / \text{neighbourhoodSize})) \geq \text{acceptThreshold})$ 
8  return  $\text{edgeDetected}$ 

```

Fig. 2. Pseudocode for the FEBD Algorithm

an edge node. Because these modifications make the original T-Fit both more cautious as well as fault-tolerant, we call this modified version *cautious*, *fault-tolerant* T-Fit or, CFT-Fit, for short.

III. CHARACTERIZATION OF DEPLOYMENT TRADE-OFFS

A. Motivation

There is a dearth of experimental studies that might serve as the basis of something like a how-to for environmental scientists wishing to deploy a sensor network for event detection. With this overall motivation, our overall goal in this paper has been to try and answer questions such as the following:

There are around 80 very reliable nodes available, with a radio range of 15 m, and there is an area of 2,500 m² to cover. Which amongst {FEBD, CFEBD, T-Fit, CFT-Fit} should be used if the minimum acceptable value for the F-measure is 0.6?

To achieve a minimum value for the F-measure of 0.8, how many nodes must be deployed to cover an area of 250,000 m² if T-Fit were to be used? And what if FEBD were used?

For this purpose, we have quantified the trade-offs between accuracy, areal coverage and sensor density that the algorithms incur. Our overarching goal has, therefore, been a practical one, viz., to inform future deployments as to what accuracy can be expected, given a desired areal coverage and such sensor density as can be afforded, by the use of the algorithms under evaluation.

B. Examples Used

We have used as the basis for defining our inputs to the algorithms under evaluation, measurements of distribution of chlorophyll obtained in Lake Fulmor (at the James Reserve in the San Jacinto Mountains, California, USA) as reported in [10]. Fig. 4 shows the interpolated data. The x-axis shows the interval [5 m, 35 m] across the transect, the y-axis show the interval [-1.0 m, -4.5 m] in depth.

We stress that we only use the data in Fig. 4 as the basis for our inputs. What we are interested in are not the values as such, let alone their meaning in nature. Instead, what is important for our evaluation is that the *event geometries* are naturally occurring, rather than idealized. As already hinted, in our comparative evaluation we have aimed to challenge

```

TFIT( $SN :: \text{list}(\text{sensorId}), \text{eventPredicate} :: \text{float} \rightarrow \text{Boolean}$ )
1  for  $s \in SN$ 
2    do in parallel over [ $s$ ]
3       $\text{sensorReading} :: \text{float} \leftarrow \text{ACQUIREREADING}()$ 
4       $\text{eventDetected} :: \text{Boolean} \leftarrow \text{eventPredicate}(\text{sensorReading})$ 
5      if  $\text{eventDetected}$ 
6        then  $\text{outgoingMsg} :: \text{triple}(\text{sensorId}, \text{float}, \text{float}) \leftarrow (s, s.\text{location}.\text{xCoord}, s.\text{location}.\text{yCoord})$ 
7          for  $n \in \text{ONEHOPNEIGHBOURHOOD}(s, SN)$ 
8            do [ $n$ ]  $\text{SEND}(\text{outgoingMsg})$ 
9       $\text{incomingMsgs} :: \text{list}(\text{triple}(\text{sensorId}, \text{float}, \text{float})) \leftarrow [ ]$ 
10     for  $n \in \text{ONEHOPNEIGHBOURHOOD}(s, SN)$ 
11       do [ $n$ ]  $\text{incomingMsgs}.\text{APPEND}(\text{RECEIVE}(n))$ 
12      $\text{edgeDetected} :: \text{Boolean} \leftarrow \text{TFITEDGEDETECTION}(\text{incomingMsgs})$ 
13     return  $(s, \text{edgeDetected})$ 

TFITEDGEDETECTION( $\text{incomingMsgs} :: \text{list}(\text{triple}(\text{sensorId}, \text{float}, \text{float}))$ )
1   $NE \leftarrow NW \leftarrow SW \leftarrow SE \leftarrow \emptyset$ 
2   $\|NE\| \leftarrow \|NW\| \leftarrow \|SW\| \leftarrow \|SE\| \leftarrow q \leftarrow 0$ 
3  for  $(n, x, y) \in \text{incomingMsgs}$ 
4    do if  $\triangleright (x, y)$  lies in the NE quadrant
5      then  $NE \leftarrow NE \cup \{(x, y)\}$ 
6         $\|NE\| \leftarrow \|NE\| + 1$ 
7    if  $\triangleright (x, y)$  lies in the NW quadrant
8      then  $NW \leftarrow NW \cup \{(x, y)\}$ 
9         $\|NW\| \leftarrow \|NW\| + 1$ 
10   if  $\triangleright (x, y)$  lies in the SW quadrant
11     then  $SW \leftarrow SW \cup \{(x, y)\}$ 
12        $\|SW\| \leftarrow \|SW\| + 1$ 
13   if  $\triangleright (x, y)$  lies in the SE quadrant
14     then  $SE \leftarrow SE \cup \{(x, y)\}$ 
15        $\|SE\| \leftarrow \|SE\| + 1$ 
16   if  $\|NE\| > 0$  then  $q \leftarrow q+1$ 
17   if  $\|NW\| > 0$  then  $q \leftarrow q+1$ 
18   if  $\|SW\| > 0$  then  $q \leftarrow q+1$ 
19   if  $\|SE\| > 0$  then  $q \leftarrow q+1$ 
20   if  $q = 0 \vee q = 4$  then return false
21   if  $q = 1 \vee q = 2$  then return true
22   if  $q = 3$ 
23     then  $\triangleright$  if  $Q$  and  $Q'$  are the diagonal quadrants and
24        $\triangleright n \in Q$  and  $n' \in Q'$  are the most Euclidean-distant nodes from  $s$  (i.e., this node) in  $Q$  and  $Q'$ , resp. and
25        $\triangleright A = \angle n s n'$  then
26       return  $(A < 180^\circ)$ 

```

Fig. 3. Pseudocode for the T-Fit Algorithm

FEED and T-Fit with inputs that are realistic in the sense just given. In this respect, consider setting the event predicate to be satisfied if chlorophyll levels lie below 20, i.e., the low concentration values in Fig. 4. These will be roughly, those drawn in darkest shades (deep blue, in colour). Note that this will give rise to a disconnected event with two sub-regions: the band at the bottom (i.e., at depths of -3.5 or more across the transect) and the dark cloud hovering just above this band.

What our experiments have shown is that depending on areal coverage and node density (i.e., depending on the *resolution* with which the sensed event is captured), both FEED and T-Fit will exhibit anomalous behaviour, as already hinted in our descriptions above of some of the modifications we made and that have given rise to CFEBD and CFT-Fit.

C. Experimental Design

In this paper, we are interested in characterizing the effect on accuracy of varying the areal coverage and the number of nodes in a deployment. We assume a radio range RR of 15 m .

We use as our dependent variable, the F-measure, i.e., the weighted harmonic mean of precision and recall (expressed in terms of true/false positives/negatives), as follows: $F = (2 \cdot TP) / ((2 \cdot TP) + FP + FN)$. For the purposes of computing the F-measure, we assume that the thickness of the edge boundary is $RR/4$, which is more stringent than the value ($RR/2$) that the literature tends to report. This means that if the algorithm calls a node an edge node that lies in the actual sensed boundary (defined to lie within a distance of $RR/8$ of the ideal sensed boundary) then it is a true positive (and so

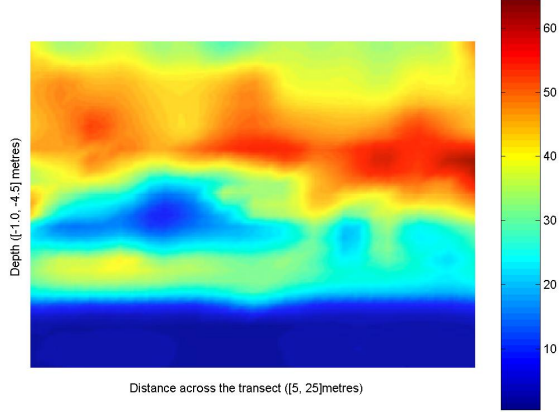


Fig. 4. Distribution of Chlorophyll, Lake Fulmor (adapted from [10])

on for all the other elements in the F-measure definition).

Areal coverage (in m^2) and the number of nodes are our independent variables.

We aim to cover four orders of magnitude in areal coverage, thus the range of areal coverage is $\{2,500m^2, 25,000m^2, 250,000m^2, 2,500,000m^2\}$. As for number of nodes, we proceed as follows. We assume that the area is covered with a grid of square cells in which nodes are deployed at the resulting coordinates. Given $RR = 15$, we chose the following range of values for the side of a square cell in the grid is $\{3m, 6m, 9m, 12m\}$. When there exists a sensor node at every point in a regular grid with n^2 cells, the number of nodes is $(n + 1)^2$.

Experiment 1 studies the accuracy for the four values of areal coverage above when there is a sensor at every point in the grid, and no sensor reading errors. We measure the accuracy of FEBD, T-Fit, CFEBD, and CFT-Fit. From the viewpoint of the deployer (but not necessarily the algorithms'), Experiment 1 is the ideal scenario. Therefore it asks the question as to what is the performance of the four algorithms in these ideal conditions.

Experiment 2 studies the accuracy for the four values of areal coverage above when there is a sensor at every point in the grid and the sensing density is constant at 20 nodes in the one-hop neighbourhood, but (a randomly distributed) 15% of the readings are erroneous. We again measure the accuracy of all the algorithms. From the viewpoint of the deployer, Experiment 2 asks the question as to what is the impact of sensing errors on the accuracy of the results.

Experiment 3 studies the accuracy for the four values of areal coverage above when there are no sensing errors, but (a randomly distributed) 15% of the points in the grid do not have sensor nodes (this could be either because the nodes have failed or because either a design decision advises against, or a physical constraint prevents, a node being deployed at that point. As a result, the sensing density varies, but is kept at a maximum of 20 nodes in the one-hop neighbourhood (i.e., the

value held constant in Experiment 2). As before, we measure the accuracy of the all the algorithms From the viewpoint of the deployer, Experiment 3 asks the question as to what is the impact of irregularity in deployment on the accuracy of the results.

Note that, because of the randomization, in the case of Experiments 2 and 3, for each setting of the independent variables, we run each algorithm 25 times and plot the average as the F-measure achieved by the algorithm.

D. Experimental Set-Up

In line with both [7] and [8], our results are simulated, i.e., the algorithms do not execute over concrete sensor network hardware. Because, in this paper, the focus is not to study the actual time, space or message complexity of the algorithms, the simulated character of our study does not, in itself, detract from the trustworthiness of our findings.

As a consequence of our simulated setting, the algorithms do not acquire readings, rather the measurements are read from data files. The data files were generated according to the experimental design described in Sec. III-C.

In order to compute the F-measure, we provide reference event geometries for each point defined by a pair of values for the independent variables. We have extracted these from the values in Fig. 4.

We implemented the four algorithms from scratch ourselves, as well as the code that extracts readings from the data depicted in Fig. 4.

E. Experimental Results

The results of Experiment 1 for each of FEBD, T-Fit, CFEBD and CFT-Fit are shown in the four plots in Fig. 6. The results of Experiments 2 and 3 are shown in the left and right plots in Fig. 7, resp..

F. Analysis and Discussion

The results of Experiment 1 confirm the intuition that, in the absence of sensor errors and with a measurement available from every point in the grid, the accuracy grows as the area covered grows, from left to right in each cluster of bars.

Notice that the number of nodes *decreases* from left to right in all of the plots. Thus, the sensing density decreases from left to right in each cluster of bars. The import of this is that those algorithms that are less robust to varying sensor density perform worse, or at least erratically for the same areal coverage. FEBD and CFEBD are in this category as is manifested by their poor accuracy when areal coverage is large but the sensing density is low. Further investigation is needed before the rather erratic behaviour of FEBD and CFEBD can be comprehensively accounted for. In contrast, both T-Fit and CFT-Fit are comparatively robust in the presence of varying sensor density.

Again as expected, in the absence of sensing errors, the introduction of a fault-suppression scheme in CFT-Fit compared with the original algorithm, is detrimental to accuracy, but not to any significant extent. This can be verified by comparing

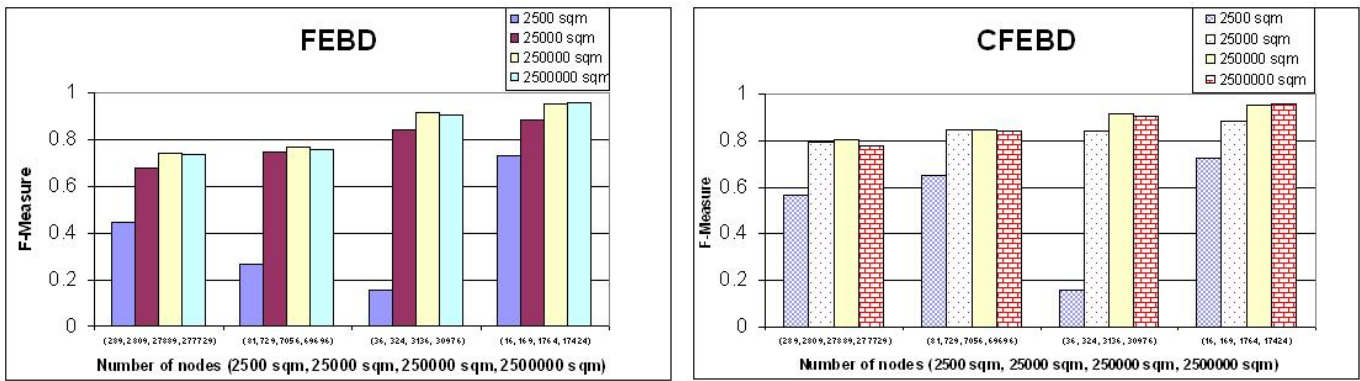


Fig. 5. Results from Experiment 1

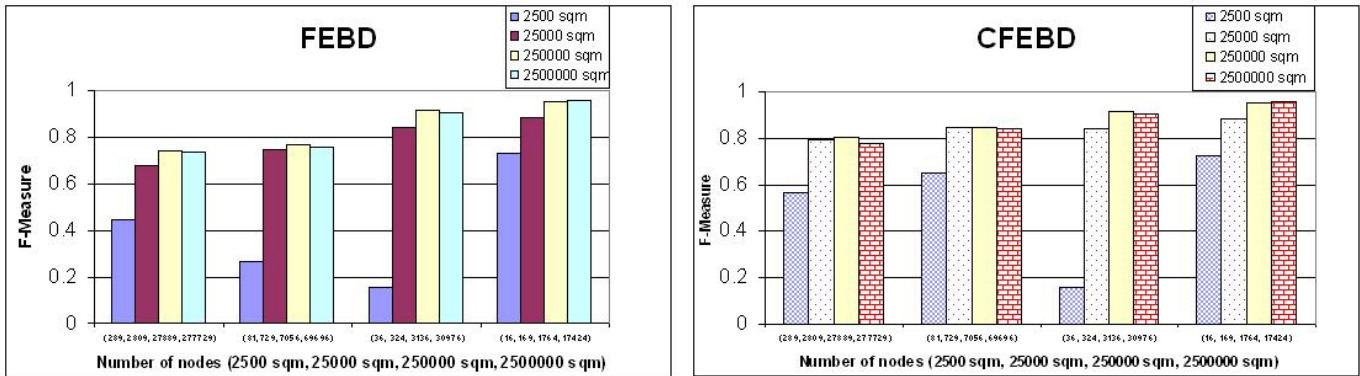


Fig. 6. Results from Experiment 1

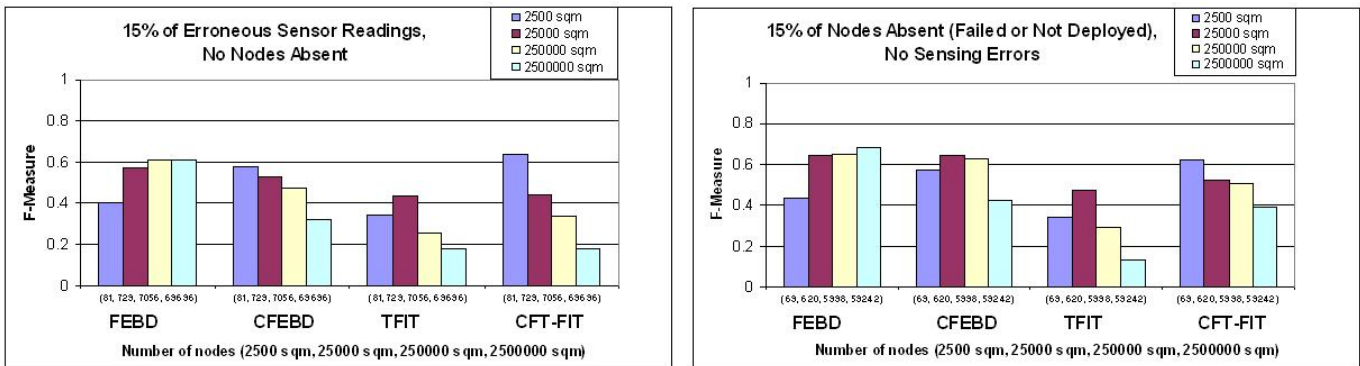


Fig. 7. Results from Experiment 2 (left) and Experiment 3 (right)

the respective plots in Fig. 6. When errors are present and when measurements are not available at some of the points in the grid, CFT-Fit is never worse than T-Fit and is often better. Admittedly, as previously remarked, this comes at the price of increased message complexity.

The results of Experiment 2 reveal that FEBD is less sensitive than the other algorithms to an increase in areal coverage. It does perform poorly for small-area networks, as shown by the fact that CFEBD, which specifically corrects for that, outperforms FEBD in the case of small-area networks. It could be argued that most deployments at this stage in

the development of sensor network technology fall in this category, in which case, as shown in Fig. 7, CFT-Fit slightly outperforms CFEBD and both outperform the algorithms as originally proposed. The same comparative analysis above holds for Experiment 3.

One lesson that can be drawn from a comparison of Experiment 1 and Experiments 2 and 3 is that, in spite of advances made since the first characterization of the **SNED** problem, current solutions remain sensitive to sensing errors and to irregular deployments. Thus, an environmental scientist would have to accept a decrease of between 0.2 to 0.4 in the F-

measure in the presence of either 15% erroneous readings or 15% failed/non-deployed nodes.

IV. CONCLUSIONS

This paper has contributed a study of two event detection algorithms, viz., FEBD and T-Fit. The study revealed shortcomings in both algorithms, which we used to develop improved versions of both, viz., CFEBD and CFT-Fit. We have quantified the trade-offs between accuracy, areal coverage and sensor density that they incur. In so doing, the paper has contributed useful information to an environmental scientist who is considering deploying a sensor network for event detection of spatially-extended physical phenomena. The results we contribute enable such a scientist to quantify the impact on accuracy of certain deployment trade-offs in localized sensor network event detection. In particular, the paper has indicated how best-of-breed solutions to the problem of sensor network event detection perform in terms of accuracy when the areal coverage and the sensor density vary. In this respect, our contributions open the way for better informed decisions in the deployment of an important class of environmental sensor networks.

For example, we can provide answers to the questions we proposed as typical in Sec. III-A, as follows:

If there are around 80 very reliable nodes available, with a radio range of 15 m, and there is an area of 2,500 m² to cover, only CFEBD and CFT-Fit deliver an F-measure above 0.6.

To achieve a minimum value for the F-measure of 0.8 over an area of 250,000 m², if T-Fit were to be used, around 3130 nodes would have to be deployed, and the same for FEBD.

As can be seen by these two examples, the study contributed in this paper can be used to inform the deployment of environmental sensor networks for the detection of events associated with natural phenomena.

ACKNOWLEDGMENT

Farhana Jabeen would like to thank the government of Pakistan for supporting her PhD studies.

REFERENCES

- [1] K. Martinez, J. K. Hart, and R. Ong, "Environmental Sensor Networks," *IEEE Computer*, vol. 37, no. 8, pp. 50–56, 2004.
- [2] J. K. Hart and K. Martinez, "Environmental Sensor Networks: A revolution in the earth system science?" *Earth-Science Reviews*, vol. 78, pp. 177–191, 2006.
- [3] K. A. Delin, "The Sensor Web: A Macro-Instrument for Coordinated Sensing," *Sensors*, vol. 2, pp. 270–285, 2002, <http://www.mdpi.net/sensors/papers/s20700270.pdf>.
- [4] K. A. Delin, S. P. Jackson, D. W. Johnson, S. C. Burleigh, R. R. Woodrwo, J. M. McAuley, J. M. Dohn, F. Ip, T. P. A. Ferré, D. F. Rucker, and V. R. Baker, "Environmental Studies with the Sensor Web: Principles and Practice," *Sensors*, vol. 5, pp. 103–117, 2005, <http://www.mdpi.net/sensors/papers/s5010103.pdf>.
- [5] G. Tolle, J. Polastre, R. Szewczyk, D. E. Culler, N. Turner, K. Tu, S. Burgess, T. Dawson, P. Buonadonna, D. Gay, and W. Hong, "A Macroscope in the Redwoods," in *SenSys*, J. Redi, H. Balakrishnan, and F. Zhao, Eds. ACM, 2005, pp. 51–63.

- [6] K. Chintalapudi and R. Govindan, "Localized edge detection in sensor fields," *Ad Hoc Networks*, vol. 1, no. 2-3, pp. 273–291, 2003.
- [7] K. Ren, K. Zeng, and W. Lou, "Fault-tolerant Event Boundary Detection in Wireless Sensor Networks," in *GLOBECOM*. IEEE, 2006.
- [8] S. Selvakennedy, "An Energy Efficient Event Processing Algorithm for Wireless Sensor Networks," in *MSN*, ser. Lecture Notes in Computer Science, J. Cao, I. Stojmenovic, X. Jia, and S. K. Das, Eds., vol. 4325. Springer, 2006, pp. 588–599.
- [9] T. C. Harmon, R. F. Ambrose, R. M. Gilbert, J. C. Fisher, M. Stealey, and W. J. Kaiser, "High-Resolution River Hydraulic and Water Quality Characterization using Rapidly Deployable Networked Infomechanical Systems (NIMS RD)," *Environmental Engineering Science*, vol. 24, no. 2, pp. 151–1459, 2007.
- [10] A. Singh, M. J. Stealey, V. Chen, W. J. Kaiser, M. Batalin, Y. Lam, B. Zhang, A. Dhariwal, C. Oberg, A. Pereira, G. S. Sukhatme, B. Stauffer, S. Moorthi, D. Caron, and M. Hansen, "Human Assists Robotic Team Campaigns for Aquatic Monitoring," *Journal of Field Robotics*, vol. 24, no. 11, pp. 969–989, 2007.
- [11] B. Krishnamachari and S. S. Iyengar, "Distributed Bayesian Algorithms for Fault-Tolerant Event Region Detection in Wireless Sensor Networks," *IEEE Trans. Computers*, vol. 53, no. 3, pp. 241–250, 2004.
- [12] T. Clouqueur, K. K. Saluja, and P. Ramanathan, "Fault Tolerance in Collaborative Sensor Networks for Target Detection," *IEEE Trans. Computers*, vol. 53, no. 3, pp. 320–333, 2004.
- [13] W. Wu, X. Cheng, M. Ding, K. Xing, F. Liu, and P. Deng, "Localized Outlying and Boundary Data Detection in Sensor Networks," *IEEE Trans. Knowl. Data Eng.*, vol. 19, no. 8, pp. 1145–1157, 2007.
- [14] G. Jin and S. Nittel, "NED: An Efficient Noise-Tolerant Event and Event Boundary Detection Algorithm in Wireless Sensor Networks," in *MDM*. IEEE Computer Society, 2006, p. 153.