2010 10th IEEE International Conference on Computer and Information Technology (CIT 2010)

Efficient Localized Detection of Erroneous Nodes (ELDEN)

Ralf Behnke, Jakob Salzmann, Stephan Simanowski, Dirk Timmermann
Institute of Applied Microelectronics and Computer Engineering
University of Rostock
18057 Rostock, Germany
{*ralf.behnke, jakob.salzmann, stephan.simanowski, dirk.timmermann*}@*uni-rostock.de*

***Abstract*—Wireless Sensor Networks (WSNs) have attracted considerable research effort in the community during the past couple of years. One of the most challenging issues so far is the extension of network lifetime with regards to small battery capacity and self-sustained operation. Endeavors to save energy have been made on various frontiers, ranging from hardware improvements over medium access and routing protocols to network clustering and role changing strategies. Only weak attention has been paid to the detection of erroneous nodes which can be also used for lifetime extension of WSNs. In contrast to some authors, regarding detection of failures in communication as error detection, we focus on faulty sensor readings. In this work we present an efficient error detection algorithm which detects erroneous nodes within the network based on neighborhood relations. The results of our algorithm enables direct reactions of concerned nodes or network parts without expensive involvement of any kind of central computing instances.**

***Keywords*-wireless sensor networks; error detection;**

I. Introduction

Recent technological advances have led to the development of tiny wireless devices, which are able to sense the environment, compute simple tasks and exchange data among each other. Interconnected assemblies of such devices, called Wireless Sensor Networks (WSNs), are commonly used to observe large inaccessible areas. A key issue to be solved before WSNs can be deployed in a wide range of applications is limited battery capacity which constrains the operational time of sensor nodes. This has been addressed by hardware manufacturers as well as research groups developing energy efficient protocols on various layers, e.g. routing, medium access and clustering. However, only few investigations have been conducted in the field of erroneous node detection. From our point of view, this topic becomes more and more important. We believe that a reliable detection of erroneous nodes has high potential of saving network energy by supporting network management with additional information.

Various stages within a sensor node's life may cause malfunction of sensors and lead therefore to faulty readings. Sensors may be incorrectly installed during manufacturing or can be impaired during rough deployment, e.g. if done by airplane. Finally sensors can fail during operation due to attrition or external influences. In particular in the field of life science automation, WSNs become more and more equipped with different kinds of sensors, i.e. physical and even chemical sensors [1]. On the one hand, especially those sensors have a higher risk of malfunction due to limited lifetime of used chemicals. On the other hand, such sensors are less power efficient than widely used ones, e.g. temperature or light sensors. If sensor nodes are equipped with such power consuming sensors, it is especially commended to use those sensors as seldom as possible. Therefore it is generally useful to detect erroneous nodes as described to adapt node and network behavior and use the additional knowledge to improve network management and increase lifetime of affected nodes and WSNs. Against this background, there is a need for a detection algorithm, suitable for a small number of erroneous nodes (about 10%), with low false positive detection rate.

As it is described in [2] erroneous nodes can be classified by the characteristic of the erroneous reading. On the one hand there are unfeasible readings and full scale readings, i.e. unfeasible in terms of sensor specification or surrounding conditions and static readings at the higher or lower measurement limit. These errors are obviously easy to detect. Another kind of error is the occurence of alternating readings as it is mentioned in [3]. In contrast to these errors this work focuses on spatial outliers, i.e. phenomena readings which differ significantly from those of neighboring nodes.

Strategies how this additional knowledge can be used to extend network lifetime, e.g. by avoiding unnecessary sensor readings and data transmissions as well as modified role assignment, are discussed more detailed in [2]. To realize such approaches sensor nodes need knowledge about their operativeness locally within the network. This requirement distinguishes our error detection approach from other approaches which detect outliers outside the network on centralized processing unit.

The remainder of the paper is organized as follows. Section II covers related work in terms of error detection approaches and related topics. In Section III our erroneous node detection approach, referred to as Efficient Localized Detection of Erroneous Nodes (ELDEN), is described. Section IV describes the simulation environment which was used to evaluate the new algorithm. The simulation results are presented in Section V, which also compares the new approach with an existing one. In Section VI a conclusion

978-0-7695-4108-2/10 $26.00 © 2010 IEEE
DOI 10.1109/CIT.2010.482

summarizes the presented work. Finally Section VII covers open tasks and future work.

II. Related Work

Many approaches concerning detection and management of errors or faults, respectively, have been published during the last years. Surveyed approaches in [4] all focus on routing, transportation and application layers as well as data dissemination. In these works, faults are seen as packet loss or total loss of (routing) nodes. In that work faulty readings are identified as a source of errors but not investigated in detail. In [5], detecting faulty nodes was identified as important research topic, but only as diagnostic tool for human networkers, further insides are not provided. The author's approach is based on comparisons between neighboring nodes, performed on a central server, to detect malicious nodes. Another example that focuses solely on connectivity is [6], which, in addition, only differentiate between alive and dead nodes. A similar problem is addressed by [7], which focuses on attacks on routing activities. In [8], SASHA is described as a self-healing architecture inspired by human lymph system. Pattern recognition techniques and neuronal networks are used to mitigate the effects of various types of faults. Therefore, non battery driven nodes, i.e. embedded computers, with higher capabilities, supported by PC class servers, are proposed to be used to monitor normal nodes. In [3], a type of self-management is presented which adapts sleep and sense cycles of neighboring nodes and is able to identify and exclude nodes with alternating readings.

In [9] Min Ding et al. describe a localized event boundary detection for WSNs. As a second aspect of their work an algorithm to detect faulty sensors is described. An important improvement over preceding work is that both algorithms accept any kind of scalar values and are not limited to 0/1 decision predicates as done in [10]. The error detection algorithm which is seen as a special case of event detection, i.e. an event limited to the erroneous node, is fully localized. By use of a kind of aggregation, phenomena readings of neighboring nodes in two-hop distance are used. Mathematical notations are used as follows. S denotes the set of all sensors in the field, while S_i denotes a single specific sensor. The actual reading of sensor S_i is defined as x_i. A subset of S that contains a node S_i and its neighbor nodes $S_{i1}, S_{i2}, \ldots, S_{ik}$ is denoted as $\mathcal{N}(S_i)$ with corresponding sensor readings $x_i, x_{i1}, x_{i2}, \ldots, x_{ik}$. Let these readings denote as $\mathcal{X}(S_i) = \{x_i, x_{i1}, x_{i2}, \ldots, x_{ik}\}$. With these notations error detection is done by checking the difference between x_i and the "center" of $\mathcal{X}(S_i)$, represented by the median which is more robust against strong outliers than the arithmetic mean.

In the first step of Ding's algorithm all nodes share its actual reading with its neighbors. In the second step each node S_i calculates the median med_i of $\mathcal{X}(S_i)$. Subsequently each node computes its difference d_i to its individual median as given in equation (1).

$$d_i = x_i - med_i \tag{1}$$

In a third step all nodes share its difference d_i among its neighborhood. Finally, Ding et al. treat a sensor S_i as faulty if its difference d_i is extreme in $D = \{d_i, d_{i1}, d_{i2}, \ldots, d_{ik}\}$, denoting the differences in its neighborhood. For this decision the mean $\hat{\mu}$ and the standard deviation $\hat{\sigma}$ are calculated in a fourth step as given in equation (2).

$$\hat{\mu} = \frac{1}{n}\sum_{i=1}^{n} d_i \tag{2a}$$

$$\hat{\sigma} = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}\left(d_i - \hat{\mu}\right)^2} \tag{2b}$$

Now, $\hat{\mu}$ and $\hat{\sigma}$ which are individual for each node are used to normalize the difference d_i as given in equation (3).

$$y_i = \frac{d_i - \hat{\mu}_i}{\hat{\sigma}_i} \tag{3}$$

Finally, in the fifth step the decision is made by equation (4) where threshold θ_{Ding} is a preselected number.

$$S_i = \begin{cases} \text{faulty} & \text{if } |y_i| > \theta_{Ding} \\ \text{normal} & \text{if } |y_i| \leq \theta_{Ding} \end{cases} \tag{4}$$

Another algorithm which is similar to Ding is described in [11]. While Ding makes indirect use of information of two-hop neighbors this algorithm uses at least two-hop information but also information of more distant nodes.

Out of the presented related work, Ding's approach fits best to our idea of erroneous nodes. In contrast to Ding [9] and Chen [11] we focus on efficiency in terms of communication and computation, i.e. less computation and one-hop communication. Therefore, our new algorithm has some similarities with the described algorithm of Ding, but offers major improvements in communication and computation.

III. Detection Algorithm

The aim ELDEN is to detect erroneous nodes in terms of local outliers. This depends on the characteristic of the phenomenon as well as on the network deployment. Algorithms like ELDEN can not be applied to phenomena which are limited to a single senor node, e.g. detections of a photoelectric barrier. In the same way sensor deployment have to be dense enough to cover laminar phenomena like temperature as well as possible gradients. As an example of such phenomena temperature is used for this paper. Although, the algorithm also covers gradients in sensor readings a consistent temperature within the sensor field is assumed. Nevertheless special cases with limited phenomena or extremely different readings are imaginable which would cause algorithms like the described one as well as the algorithm of Ding to fail. In some cases also an initial run of the error detection can be assumed to find erroneous nodes under normal conditions, i.e. no features which could lead to a faulty detection. Due to the fact, that the erroneous node detection algorithm described by Ding et al. matches the challenges which we defined for an useful erroneous node detection algorithm, it was an inspiration for a new approach, referred to as ELDEN.

There are two major drawbacks of Ding's algorithm which are removed by the new algorithm. First drawback is the demand on communication. Ding's algorithm contains two steps of high communication. Second drawback of Ding's algorithm is the computational demand. As illustrated in Section II, final decision whether a node is faulty or not is based on difference d_i, mean $\hat{\mu}$ and standard deviation $\hat{\sigma}$. Therefore, a number of divisions and root calculations as described in equations (2) and (3) have to be performed on constraint nodes for local decision.

In contrast to Ding's algorithm ELDEN strictly pursues the idea of median as a substitute of mean which is more robust and can be easily determined. Median is only one special case of the general concept of quantiles. For getting q-quantiles the given data set have to be sorted first and then split into q essentially equal-sized data subsets. The q quantiles are denoted as the data values marking the boundaries between consecutive subsets.

While Ding only uses the second quartile, i.e. the second 4-quantile, ELDEN also facilitates the first and the third quartile. The difference between first quartile and third quartile, known as Interquartile Range (IQR), is used to rate the difference between the node's reading and the calculated median of neighboring nodes.

First step of ELDEN is equal to Ding's algorithm, i.e. each node promotes its own actual reading. After this step all nodes are informed about the actual readings of its neighboring nodes in one-hop distance, i.e. $\mathcal{X}(S_i)$ is determined. In contrast to Ding this is the only communication needed in ELDEN.

In a second step each node S_i calculates median med_i and IQR IQR_i with the help of $\mathcal{X}(S_i)$. Difference d_i is calculated as given in equation (1).

As the third step y_i is calculated as normalized difference as given in equation (5).

$$y_i = \frac{d_i}{IQR_i} \tag{5}$$

Last step is the decision itself, which bases on a threshold θ_{ELDEN} and is performed as given in equation (6).

$$S_i = \begin{cases} \text{faulty} & \text{if } |y_i| > \theta_{ELDEN} \\ \text{normal} & \text{if } |y_i| \leq \theta_{ELDEN} \end{cases} \tag{6}$$

Without additional communication and costly communication, ELDEN draws its conclusion with about half the complexity, used by Ding.

IV. Simulation environment

To verify performance and proper work of the described erroneous node detection, it has been tested in a MATLAB® based simulation. For simulation purpose a two dimensional quadratic sensor field is defined with given field size. A certain number of sensor nodes is randomly deployed in that field. In the next step an individual temperature reading is assigned to each sensor node. Therefore a fixed temperature value and a corresponding standard deviation has been chosen in the simulation. This value is assigned to unaffected nodes with normal distributed deviation. A further simulation parameter defines a percentage of nodes which are going to produce erroneous readings. These nodes are randomly chosen. Their reading values provide a deviation to normal nodes, referred to as *fault difference*, which has been chosen separately and is assigned with the same normal distributed deviation.

Intuitively known, success of a detection algorithm depends on the number of neighboring nodes which can be used for comparison. Obviously this parameter is influenced by the number of nodes, deployed in the whole field and the communication range, assigned to each node. Although both parameters should be seen in relation to each other, they are separately adjustable within the implemented simulation. Due to the fact that the detection process does not depends on signal strength between neighboring nodes unit disc graph model was used for modeling transmission range. Each set of parameters was used for 200 randomized simulations.

As described in Sections II and III, Ding's algorithm as well as the presented new approach rely on a certain threshold. Although in both cases thresholds have similar meanings, they can not be directly compared. Nevertheless simulations have been done with same threshold for both algorithms. For evaluation simulations with different thresholds can be compared with each other without penalize one of the algorithms.

To rate the new algorithm and for comparison with Ding's algorithm the simulation produces several results. True positives indicates the percentage of correctly identified erroneous nodes. In contrast, percentage of unaffected nodes, wrongly detected as erroneous are given as false positives. For better comparison the average number of neighboring nodes is determined. An overview of input parameters of the simulations is given in table I.

V. Results

As described above, a variety of parameters with impact to detection results have been studied. Due to the fact that it is

Table I
INPUT PARAMETERS OF SIMULATION

parameter	value
number of runs	200
length of field [m]	100
number of nodes	500, 1000, 1500
threshold θ	1..4
normal reading [°C]	20 ± 2
fault difference [°C]	5..50
faulty nodes [%]	5..50
communication range [m]	5..50

impossible to illustrate all results, gathered from simulations, it was a task to find appropriate parameter values.

A. Thresholds

First task was to determine thresholds for both described algorithms, which work best for the targeted application. Figure 1 shows percentage of true positive detected faulty nodes over percentage of faulty nodes. The number of nodes used for that simulation was set to 500 and communication range was set to 20 m, resulting in about 53 nodes in a one-hop neighborhood. Fault difference was set to 20 °C. It is illustrated, that the algorithm of Ding detects most faulty nodes if threshold is set to 1 ($\theta_{Ding} = 1$). It is also shown that Ding detects less faulties if higher thresholds are used. Similarly, it is illustrated that ELDEN also detects more faulty nodes if a small threshold is used. In comparison to Ding, threshold has only weak influence on true positives, which is shown in the fact that all studied variants of ELDEN perform better than Ding with threshold 2 or higher.

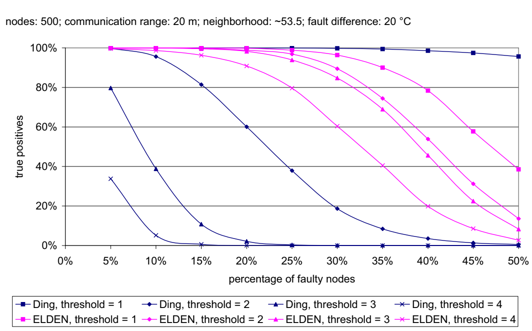

Figure 1. True positives over percentage of faulty nodes and various thresholds

To find appropriate thresholds, also percentage of false positives have to be taken into account. Using the same parameter set as in figure 1, percentage of false positives is illustrated in figure 2. It is shown that the number of false positives is higher if only few faulty nodes exist. Also more faultless nodes are detected as faulty if a low threshold is chosen.

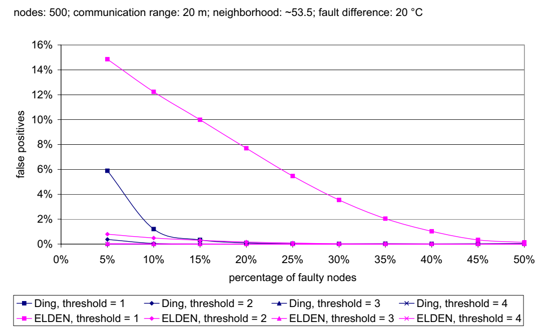

Figure 2. False positives over percentage of faulty nodes and various thresholds

Due to the presented results, a threshold of 1 is chosen as optimum for Ding ($\theta_{Ding} = 1$), providing best results for true positives and good results for false positives. Although threshold 2 is better regarding false positives, it is not preferred due to the bad performance regarding true positives. Optimal threshold for ELDEN is chosen as 2 ($\theta_{ELDEN} = 2$), which provides excellent results concerning false positives and good results concerning true positives. Especially for the targeted case of a small number of faulty nodes, i.e. less than 25%, ELDEN with threshold 2 provides good results. Therefore, these thresholds are used in further comparisons.

B. Number and magnitude of faulty nodes

To analyze the influence of number and magnitude of faulty nodes, both parameters are varied in figure 3, which shows true positives for the algorithm of Ding. Once again a communication range of 20 m is used. As expected, faults will be detected better, if the difference between faulty readings and faultless readings is high. The influence of this parameter have to be seen in relation to the standard deviation, used for faultless and faulty readings, which was set to 2. Furthermore, it is shown that detection is more reliable if only few nodes are faulty.

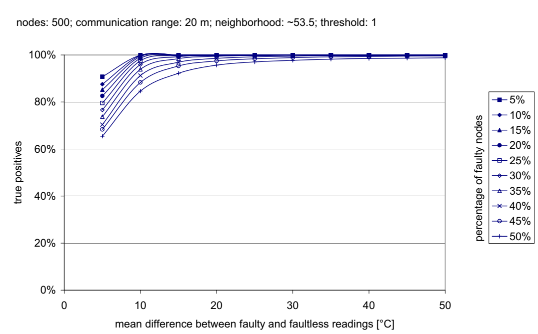

Figure 3. Ding: True positives over fault difference and percentage of faulty nodes

A similar behavior is illustrated in figure 4, concerning

ELDEN. Only few faulty nodes can be detected if the reading differs only marginal from faultless readings. Percentage of faulty nodes instead affects ELDEN much more than Ding. Nevertheless ELDEN detects more than 96% of faulty nodes if the percentage of faulty nodes is less than 25% and fault difference is 15 or higher.

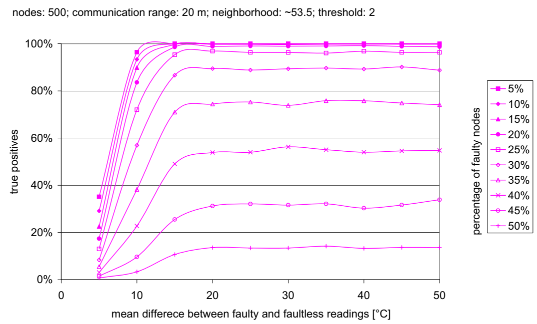

Figure 4. ELDEN: True positives over fault difference and percentage of faulty nodes

While Ding outperformes ELDEN for general cases, concerning true positives, it is worse concerning false positives as depicted in figure 5. Especially in presence of few faulty nodes a high number of faultless nodes are detected as faulty. Also fault difference highly influences false positives.

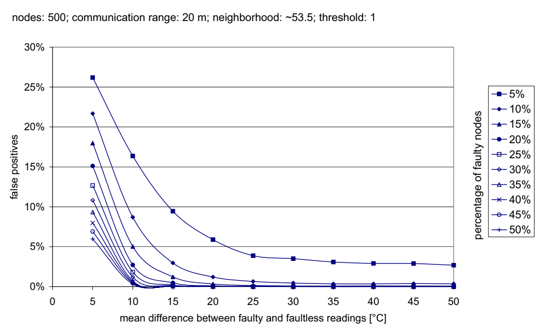

Figure 5. Ding: False positives over fault difference and percentage of faulty nodes

In contrast, the number of false positives, provided by ELDEN, is extremely low, even if only few faulty nodes exist. This is illustrated in figure 6. Additionally, it is shown that fault difference has no influence on false positives, using ELDEN.

C. Communication range and number of nodes

Performance of both algorithms depends on the number of nodes, used to detect a faulty node. This parameter, referred to as neighborhood, is influenced by communication range and number of nodes, deployed per unit area. While simulations have shown that communication range and number of nodes are equivalent, influence of communication range is illustrated only. For the following illustrations, found

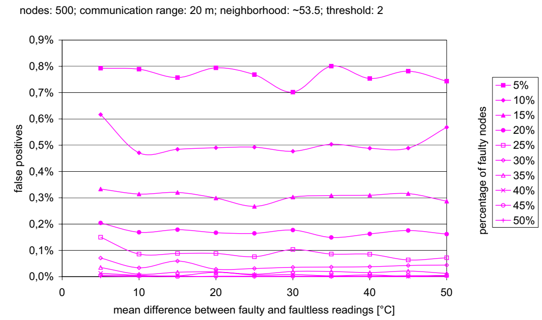

Figure 6. ELDEN: False positives over fault difference and percentage of faulty nodes

thresholds, i.e. $\theta_{Ding} = 1$ and $\theta_{ELDEN} = 2$, and 500 nodes are used. Difference between faultless and faulty readings is set to 20 °C.

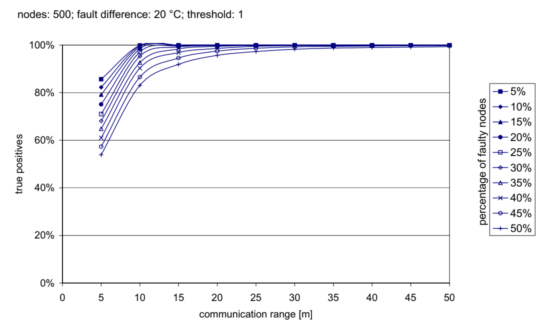

Figure 7. Ding: True positives over communication range and percentage of faulty nodes

Figure 7 shows that true positive detection of faulty nodes, using Dings algorithm, depends on the communication range. For a good detection of more than 97%, presuming not more than 25% of faulty nodes, a communication range of 10 m is sufficient. To achive similar results using ELDEN, a communication range of at least 20 m is needed, illustrated in figure 8.

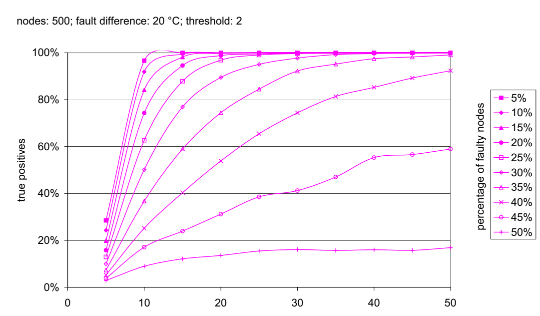

Figure 8. ELDEN: True positives over communication range and percentage of faulty nodes

As expected from former results, concerning false positives, Ding is outperformed by ELDEN. Figure 9 shows relation between communication range and false positives for various percentages of faulty nodes using Ding. It is illustrated that having about 5% faulty nodes in the network, Ding provides a false positive rate of at least 2%, using a high communication range. Having 10% faulty nodes, a communication range of 25 m is sufficient to achieve less than 1% false positives. For a higher percentage of faulty nodes a communication range of at least 15 m is needed to get less than 1% false positives.

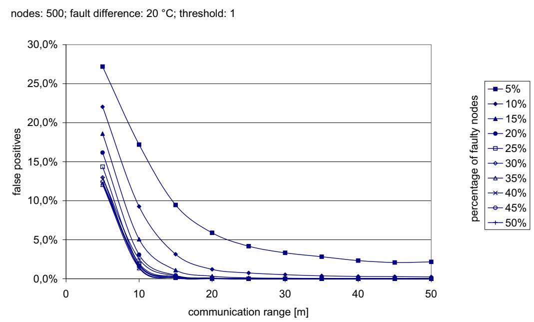

Figure 9. Ding: False positives over communication range and percentage of faulty nodes

In contrast to Ding, ELDEN achieves less than 1% false positives even for small communication ranges and a small percentage of faulty nodes. Figure 10 shows that false positive rate is slightly influenced by communication range. But compared to Ding, concerning false positives, ELDEN performs nearly independent from communication range and outperforms the algorithm of Ding, providing good results even for a low percentage of erroneous nodes. Outliers at 5 m are caused by a very small neighborhood of 4.7 nodes.

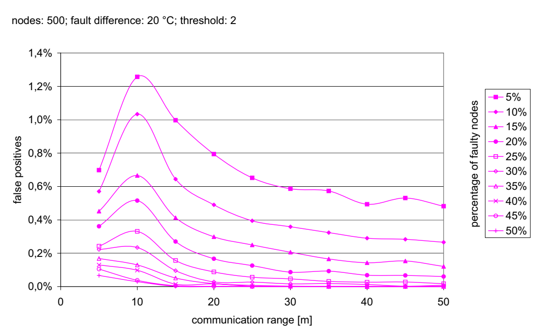

Figure 10. ELDEN: False positives over communication range and percentage of faulty nodes

For comparison, figure 11 shows true positives over the mean number of neighbors per node. The number of faulty nodes is set to 10% providing a fault difference of 20 °C. It is illustrated that Dings algorithm, using a threshold of 1, performs slightly better and achives good results of more than 99% if more than 12 nodes constitute a neighborhood. In contrast, ELDEN, using a threshold of 2, needs a neighborhood of about 30 nodes to achive similar results. Beside the differences in calculation one reason for that behavior is that Ding achieves an indirect neighborhood with twice the radius and a quadruple number of nodes, reasoned by its second communication phase which covers information related to each nodes neighborhood.

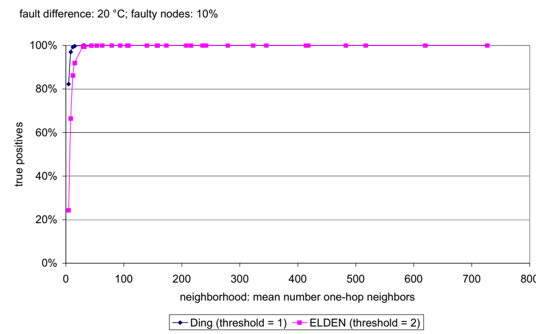

Figure 11. True positives over mean size of neighborhood

Oppositional results have been found for false positives, as illustrated in figure 12. While ELDEN produces less than 1% false positives from the beginning, i.e. a neighborhood of about 5 nodes, Ding undergoes this threshold for the first time using at least 63 nodes within a neighborhood. In particular, for the given parameters, ELDEN detects as good as Ding but with less false detections if a nodes neighborhood is about 30 to 60 nodes in size.

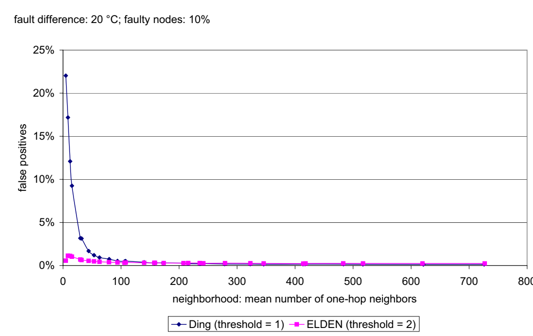

Figure 12. False positives over mean size of neighborhood

D. Cost and Complexity

Cost and complexity, caused by communication and computation are not part of the performed simulations. Nevertheless, it can be stated that ELDEN outperforms Ding in both aspects. First of all, ELDEN uses only one communication phase, which is equal to the first communication part, used by Ding. So communication is significantly reduced compared to Ding. Also computation is significantly reduced as well. On the one hand, Ding orders readings of neighboring nodes to determine the median as its first computation, which is followed by a second computation, costly determining mean and standard deviation. In contrast, ELDEN needs only the first part of computation. In addition to Ding, ELDEN determines quartiles by picking the right values out of the ordered list. While the values are still ordered from determining the median, no additional ordering is needed. Therefore, at most four additional additions and four additional divisions are needed to determine the two remaining quartiles, getting the right index and determining the mean if necessary.

VI. Conclusion

The presented new approach for detection of erroneous nodes outperforms the existing algorithm of Ding in terms of communication and computation. Furthermore, the new algorithm, referred to as ELDEN, beats Ding in terms of false positive detections. Concerning true positive detections, ELDEN performs good for small numbers of faulty nodes but can be still outperformed by Ding, especially if a small neighborhood is used.

Regarding the targeted aim of detecting sensor nodes with erroneous readings to stop propagating erroneous values, ELDEN is particularly suitable, cause less proper working nodes become inadvertently stopped and a large number of faulty nodes become successfully detected assuming that typically 5% to 10% but at most 25% of all nodes are faulty. The detection works better if erroneous readings highly differ to faultless readings.

VII. Future Work

There are several aspects which will be investigated in future work. One of the most interesting tasks is to quantify potential of energy saving and lifetime extension with the help of realistic simulations including aspects of routing and an appropriate energy model. Beside the use of error detection, the detection process itself can be improved. By exploiting cluster structures, which are widely used in WSNs, detection of erroneous nodes can become more efficient. A clustering approach which is particularly suitable is 2-MASCLE [12]. This approach ensures that each node in a cluster is able to observe the same phenomena, i.e. a phenomenon within the cluster, which allows the cluster members to control each other. Also adjacent clusters are able to control themselves because a phenomenon within a cluster can also be detected by at least one neighboring cluster.

An interesting point is to see how erroneous node detection reacts if nodes, once detected as erroneous, are not taken into account within a further detection phase. With regard to the presented simulation results, it is supposable, that a fraction of faultless nodes will be detected as erroneous within each detection, even if no erroneous are within the network. It would be a challenge to find an appropriate criterion to prevent the network from such a graceful degradation.

Acknowledgment

This work was supported by the German Research Foundation under grant number BI467/17-2 (keyword: Geosens2)

References

[1] R. Behnke, F. Golatowski, K. Thurow, and D. Timmermann, "Wireless sensor networks for life science automation," in *International Forum Life Science Automation*, 2007.

[2] R. Behnke, J. Salzmann, D. Lieckfeldt, K. Thurow, F. Golatowski, and D. Timmermann, "Exploiting malicious node detection for lifetime extension of sensor networks," Jan. 2009, pp. 1–2.

[3] P. Boonma, P. Champrasert, and J. Suzuki, "A biologically inspired architecture for self-managing sensor networks," *Sensor and Ad Hoc Communications and Networks, 2006. SECON '06. 2006 3rd Annual IEEE Communications Society on*, vol. 3, pp. 734–739, Sept. 2006.

[4] Paradis, Lilia, Han, and Qi, "A survey of fault management in wireless sensor networks," *Journal of Network and Systems Management*, vol. 15, no. 2, pp. 171–190, June 2007.

[5] C. Jaikaeo, C. Srisathapornphat, and C.-C. Shen, "Diagnosis of Sensor Networks," in *IEEE International Conference on Communications (ICC 2001)*, Helsinki, Finland, June 2001.

[6] X. Du, M. Zhang, K. Nygard, M. Guizani, and H.-H. Chen, "Distributed decision making algorithm for self-healing sensor networks," *Communications, 2006. ICC '06. IEEE International Conference on*, vol. 8, pp. 3402–3407, June 2006.

[7] W. R. P. Junior, T. H. de Paula Figueiredo, H. C. Wong, and A. A. F. Loureiro, "Malicious node detection in wireless sensor networks," *ipdps*, vol. 01, p. 24b, 2004.

[8] T. Bokareva, N. Bulusu, and S. Jha, "SASHA: Toward a Self-Healing Hybrid Sensor Network Architecture," *Embedded Networked Sensors, 2005. EmNetS-II. The Second IEEE Workshop on*, pp. 71–78, May 2005.

[9] M. Ding, D. Chen, K. Xing, and X. Cheng, "Localized fault-tolerant event boundary detection in sensor networks," *INFOCOM 2005. 24th Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings IEEE*, vol. 2, pp. 902–913 vol. 2, March 2005.

[10] B. Krishnamachari and S. Iyengar, "Efficient and fault-tolerant feature extraction in wireless sensor networks," *Information Processing in Sensor Networks*, pp. 488–501, 2003.

[11] J. Chen, S. Kher, and A. Somani, "Distributed fault detection of wireless sensor networks," in *DIWANS '06: Proceedings of the 2006 workshop on Dependability issues in wireless ad hoc networks and sensor networks*. New York, NY, USA: ACM, 2006, pp. 65–72.

[12] J. Salzmann, R. Behnke, D. Lieckfeldt, and D. Timmermann, "2-MASCLE - A Coverage Aware Clustering Algorithm with Self Healing Abilities," *Intelligent Sensors, Sensor Networks and Information, 2007. ISSNIP 2007. 3rd International Conference on*, pp. 739–744, Dec. 2007.