Classification of Faults in Sensor Readings with Statistical Pattern Recognition

Valentina Baljak†*, Kenji Tei* and Shinichi Honiden†*
†*University of Tokyo*
**National Institute of Informatics*
Tokyo, Japan
Email: {valentina, tei, honiden}@nii.ac.jp

***Abstract*—In wireless sensor networks, frequent faults are caused by general characteristics and the direct exposure to the environment. Accumulation of these faults can lead to the progressive decrease of reliability and accuracy of sensor readings. We focus on detection and classification of faults within sensory data independently of the underlying cause. We propose a complete and consistent fault classification based on two aspects. The first aspect is continuity and frequency of the occurrence, and the second is the existence of observable and learnable patterns. Given that modeling of faults prior to the detection is a fundamental process, we address it with statistical analysis and theoretical approach. We rely on centralized and straightforward detection methods using neighborhood vote. For the full classification phase, we propose the use of statistical pattern recognition with a priori modeling of faults. Current results show that this method works comparatively well when applied to collected data in data centric dense wireless sensor network.**

***Keywords*-Fault Tolerance; Wireless Sensor Networks; Pattern Recognition; Faults Classification**

I. INTRODUCTION

Wireless sensor networks (WSN) are complex systems that consist of a large number of small, cheap devices. We deploy a WSN to collect and process data in order to understand the behavior of a monitored entity. Often, the network needs to perform demanding scenarios in a harsh environment. The significantly lower intrinsic reliability of sensors and actuators than that of integrated circuits in enclosed packaging comes from their direct contact with the environment [1].

Ultimately, the goal of WSN is to provide accurate data about monitored phenomena efficiently over the maximum possible period. Even when perfectly calibrated in the beginning, network will accumulate a number of faults over the time. This leads to the shortening of its effective lifetime, defined as the time of operation in which network reliably provides accurate data.

This research focuses on detecting faults that occur in sensor readings within dense data-centric WSN. In particular, it keeps a focus on the pattern of occurrences as they appear in readings of each sensor node. These observations are independent of the underlying cause of the fault. In this case, a fault is the manifestation of erroneous reading within the data, regardless of the underlying cause for this reading. In this case, we rely on the neighborhood vote and spatial and temporal correlation of the readings, while there is no correlation amongst faults developing on each node.

One of the challenges for fault detection is a proper modeling of faults. Detection of faults relies on these models for the accuracy. We attempt to provide initial models for the detection and classification of faults as they can be observed in collected readings. Frequency and continuity of occurrence is the basis for the proposed classification of faults. The aim is to provide complete and consistent classification, independent of the underlying causes for faults, such as errors in calibration or packet loss.

Amongst challenges that WSN face is the quality of service, where the most important aspect is the amount and the quality of the information that can be extracted at any given sink about the observed object or area. Fault tolerance at all levels of the network is a necessary trait of the required QoS, especially the quick recovery from a fault.

Lifetime of a network is another crucial figure of merit. It depends largely on the energy consumption. For this reason, fault tolerance mechanisms have to find a way to balance the cost of communication and computation. One straightforward definition of a lifetime is given as the time before a first node runs out of energy and stops transmitting [2]. However, data accuracy and reliability might fail long before that, due to accumulated faults. Goal of this research is to use fault mechanism to prolong the effective lifetime of the network, defined as a time in which network provides reliable and accurate data.

Analysis of related work on fault tolerance in WSN is discussed in Section II. Details of the proposed approach and results are presented in Section III, followed by conclusion, discussion and future work in Section IV.

II. FAULT TOLERANCE IN WIRELESS SENSOR NETWORKS

Fault tolerance is a fundamental requirement for efficient and reliable operation of WSN. In order to provide a quick recovery, fault latency must be low. This means that a network must be able to eliminate the effect of the fault in a short time. Some classic approaches rely on neighborhood vote techniques or data redundancy, and the existing literature covers them extensively [2], [3]. Apart from these

 ISBN: 978-1-61208-207-3

techniques, there is a trend towards using intelligent data fusion and machine learning to provide more flexible fault recovery, for example, Shell et al. propose fuzzy data driven fusion with statistical process control [4].

For example, Takruri et al. propose SVR-UKF-IMM framework for auto calibration [5]. It is an efficient method for fault detection and error correction, but it requires a repetition in each round of data collection. The presented method enables each sensor to predict a correct reading based on information from the previous cycle. Although this approach is flexible, it leaves the network unaware of the occurrence and location of a fault. The sink will never receive an actual reading from the sensor, only a corrected one. With the centralized approach, the network would be aware of the exact node where the fault has occurred.

Koushanfar et al. give a detailed analysis of fault tolerance [1]. Amongst other requirements, they state that it is preferable to develop approaches that require little additional computation regardless of any additional communication requirements as the answer to the common dilemma about the tradeoff between computation and communication. We believe that centralized approach to the fault tolerance corresponds to this requirement since it does not require additional communication, and it ensures enough computational power. Centralized approach is supported by Ni at al. [6]. Also, they point out that any value exceeding a high value threshold is not necessarily a fault. Assumption that nodes need only to be correlated and have similar trends is not enough. Authors emphasize the role of modeling of data since it might provide the basis for comparison if there is a lack of other references. Consistent initial classification of faults can provide a sufficient basis for better modeling of faults initially, as well as learning models from experience. Sharma et al. examine the prevalence of sensor faults in different datasets obtained from real deployments [7]. Authors focus on several types of transient errors and evaluate several different methods for the fault detection. Their conclusion is that the frequency of appearance varies. It ranges from 0% - 20% for different types of errors and different datasets. In the end, authors conclude that the influence to the overall functionality of the network is significant regardless of the frequency of faults. Also, they believe that online handling of faults is an important task. Yao et al. in [8] support this opinion and they propose a straightforward approach to the online detection using time series.

Approach presented here addresses fault tolerance in data centric, densely deployed network with assumed spatial and temporal correlation. For comparison, literature offers some interesting examples of different approaches in different settings, as seen in Tiwari and Thai, who are concerned with providing a fault tolerant virtual backbone as the routing infrastructure [9]. Dual cluster cooperating scheme for data gathering network is proposed by Huang et al. [10]; with the advantages of reduced data loss and a small communication overhead. Also, this scheme can detect both, link faults and node faults. Interesting application and results from a real-time deployment in the avionics is reported by Alena [11].

Fault tolerance is an essential part of reliable sensor network operation. It is obvious that different types of networks and different applications pose varied requirements. To provide a satisfactory fault tolerance scheme, any approach should concern itself with all of the main aspects of fault tolerance, fault models, detection and diagnosis and resiliency mechanisms. Classical techniques, like neighborhood vote, provide a reliable, easy to adapt approach for detection. However, machine learning techniques provide more flexibility for recognizing the type of faults and handling them accordingly. Also, these methods provide a critical difference in terms of early discovery of faults that occur continually over the prolonged period. This significantly affects capability of the network for the quick recovery.

III. Classification with Statistical Pattern Recognition

Part of the fault tolerance recognition and resiliency mechanisms are various criteria for fault classification. In general, error is a manifestation of a fault inside of a program that can occur either at the fault site or at some distance. Together with the fact that each level of abstraction has its own types of faults, current literature covers classification of faults based on various criteria.

Faults can occur in different layers of WSN. If we look at the location and cause of faults, most commonly they appear at physical layer, since sensors and actuators are most prone to malfunctioning. Faults that can occur at this level can besuch as physical layer, where we can have communication faults or hardware malfunctions and energy supply problems. Since sensors and actuators are the most vulnerable components, at this level we can classify faults as:

- calibration systematic errors,
- random noise
- complete malfunctioning

Calibration errors are probably a key source of all faults since they can manifest themselves as a bias or a drift throughout the lifetime of the sensor node.

In the middleware, focus moves towards data aggregation, filtering and sensor fusion, all of which are tasks dependent on accurate and reliable sensor readings. However, it is difficult to provide a fault tolerance at the level of a single sensor node in an economic way. Addressing faults at the application level is efficient, but requires a customized way of addressing each issue. On the other hand, this provides flexibility to address faults regardless of the resource or level in which faults appear [1].

If we take a different look at faults, for example, how often and how long they appear, they can be classified based on the time and persistence as **permanent faults**,

 ISBN: 978-1-61208-207-3

continuos and stable in time; **intermittent faults**, occasional manifestation due to unstable characteristics and **transient faults**, reflective of temporary environmental impact.

Ni et al. give extensive taxonomies of faults that cover definitions, causes, duration and impact of faults [6]. For example, calibration fault can be detected as the fault where the reported value is offset in some way from the ground truth. Bias and drift are primary causes of these faults. This fault is persistent, and it remains present throughout the deployment. However, data should not be discarded, since proper calibration formula can correct them. As opposed to this, faults occurring at hardware or connection level render data useless and should be disregarded. In a similar manner, spikes do not provide a meaningful information, and data should be disregarded. The difference is that spikes might occur only sporadically, while hardware error is persistent. It would make sense to keep the first node active, while the second should be deactivated.

We believe that the capability to distinguish between different types of faults would provide WSN with increased flexibility in handling faulty nodes. This flexibility can be achieved by learning from observations. In this way, a network could improve its behavior based on the study of its own experience. Given the goal in this work to recognize and classify different types of faults that occur on sensor nodes, we have chosen to focus on statistical learning, more specifically on statistical pattern recognition [12], [13].

The review of existing literature led us to focus on reliability of data-centric, densely deployed sensor networks. In this sense, we have formed a classification of faults in sensor readings independent of the cause of fault or the location of occurrence.

A. Fault Classification

In this work, we focus on faults that can be observed in readings through the effect they produce in data. This is a data-centric, diagnostic approach. Data features are statistical in nature, and a confident diagnosis of any single fault may require the use of more than one of those features. In dense, data-centric, networks readings are spatially and temporally correlated, so statistical patterns of readings can be used to identify and describe faults.

After analyzing existing classifications and underlying criteria, we have chosen to determine if faults can be recognized based on the pattern of behavior that they leave in the data. We propose a complete and consistent classification of faults in sensory data in terms of:

- Continuity of the occurrence
- Frequency of the occurrence
- Observable and learnable pattern

Criteria we have chosen draw from existing experience, and generalize and abstract existing criteria. We believe that a classification based on these criteria is flexible and applicable to a wide range of sensor readings. At the same time, it decouples models of faults from physical characteristics of a network and from the environment. Underlying cause of the error does not affect this classification. The focus of this work is on the pattern of fault occurrences on each sensor node.

If a sensor reading is represented with $r_i + \varepsilon_i$, where ε_i is a fault, we can define fault classification as follows:

- **Discontinuous** - Fault occurs from time to time, occurrence of ε_i is discrete.
  - **Malfunction** - Frequent occurrence of faulty readings, $\varepsilon_i > \tau$, where τ is threshold frequency. Also, there is no observable pattern in the fault occurrences.
  - **Random** - Infrequent occurrence of a faulty readings, $\varepsilon_i \leq \tau$.
- **Continuous** - After the certain point in time, a sensor returns constantly inaccurate readings, and it is possible to observe a pattern in the form of a function:

$$\varepsilon_i = f(t, [\alpha_1, \alpha_2....])$$

  - **Bias** - The function of the error is a constant, $\varepsilon_i = const$. This can be a positive or a negative offset.
  - **Drift** - The deviation of data follows a learnable function, such as polynomial change

$$\varepsilon_i = \alpha_1 * \varepsilon_{i-1}^n + \alpha_2 * \varepsilon_{i-1}^{n-1} + ...\alpha_0$$

Figure 1 illustrates the main concept of this classification. Knowing the type of the error provides a network with flexibility in handling faulty nodes appropriately, according to their type. Random faults can be smoothed out through data fusion while malfunctioning nodes can be switched off. Continuous faults are more interesting and challenging to handle in this case. This classification can provide a basis for the use of hypothesis finding techniques in order to learn a function of the fault and apply that function (model) to subsequent readings. Furthermore, models we learn can be used to update a priori set models of expected faults.

The classification process is a part of a possible framework illustrated in the Figure 2. This framework addresses a full cycle of modeling-detection-resiliency mechanism. However, this full cycle is out of the scope of this paper.

B. Proposed Method and Results

This work deals with the discovery of faults in sensor readings of data-centric WSN. One of the most significant metrics is accuracy, defined as the difference between resulting value and true value [2]. We rely on a neighborhood vote, keeping the assumption of spatial and temporal correlation of measurements. This means that we expect nodes in the same area to measure the same phenomena at the same time. The assumption holds for data-centric densely deployed WSN, as opposed to the use of saving mechanisms that turn off the network interface opportunistically which leads to intermittent connectivity and the formation of a Delay/Disruption

 ISBN: 978-1-61208-207-3

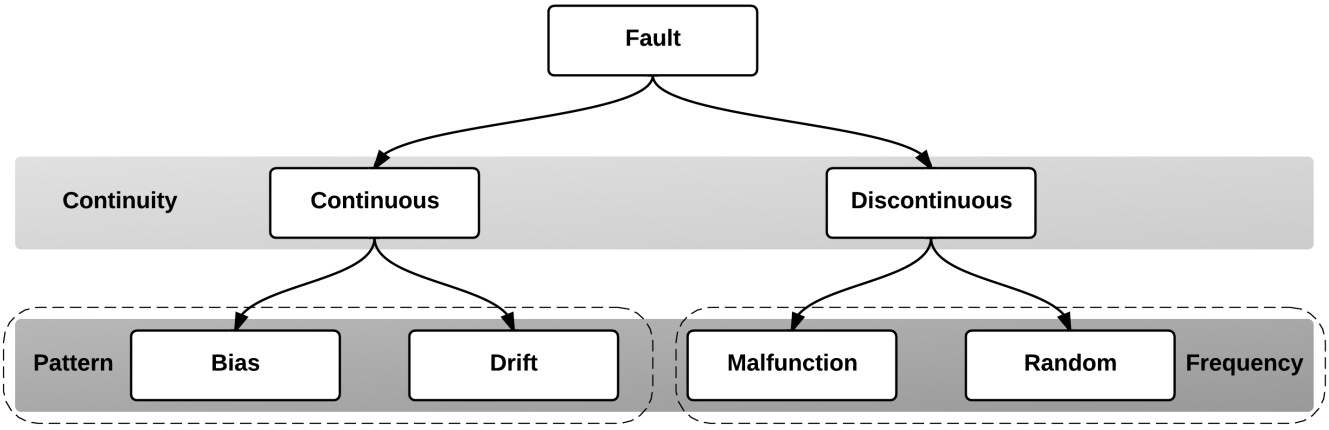


Figure 1. Classification of faults based on frequency, continuity and an observable pattern

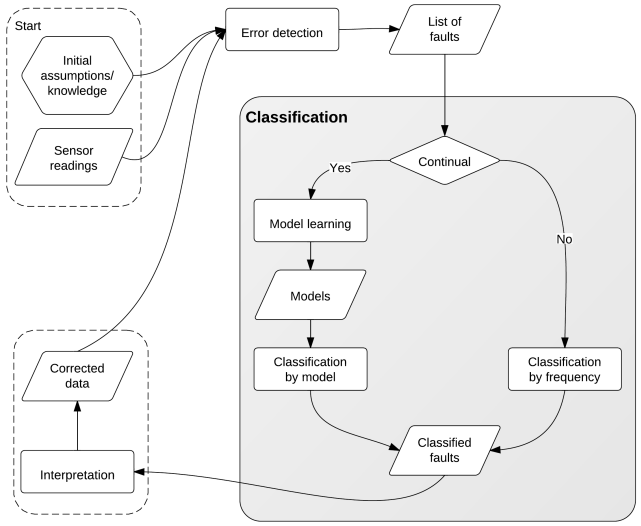


Figure 2. The process of fault handling with automated framework

Tolerant Network (DTN) [14]. An important characteristic of these networks is the lack of a simultaneous path between source and destination, while existing literature on routing in DTNs assumes that data packets are independent and uncorrelated with one another.

For the proposed method, it is necessary to assume the existence of a good connectivity between the nodes in the same area. Nodes are capable of clustering based either on the recognition of a nearby neighbors or an in-built detailed knowledge about the network layout, both physical and relational.

Every fault tolerant system has to decide between connectivity and communication overhead. Some of the existing work shows good results with on-node fault correction, with the benefit of flexibility and a quick response. On the other hand, this approach can cause a significant loss of information about a node's behavior. The network would always have to rely on corrected readings based on the prediction instead of on actual readings. In the centralized approach, the network becomes aware of the behavior of each node. When the network is capable of recognizing the type of fault, it can correct the readings and handle the node accordingly. Centralized approach does not cause any increase in communication overhead and provides sufficient resources for potential heavy computation requirements.

For the experiments, we have used the well known Intel Berkely laboratory dataset [15]. This set provides data on time and epoch of measurements, temperature, humidity, light and voltage. Dataset was split based on the location of sensor nodes, for example, a group of nodes with id 44 - 48.

In the first step, we analyze collected data in the time series to discover any discrepancies that might indicate the existence of faults. We are using statistical features of data for this purpose. For the initial calculations we rely on calculating the median, $\mu = \{r_{ij}\}_{\frac{1}{2}} = \tilde{r}$, of the group for the each epoch and chosen measurement, e.g. temperature. Median is chosen over the average as a more robust metrics. It is less likely to change drastically if extreme spikes in measurements occur on some of the nodes.

The difference between the reading and the median is calculated for each node, within the given tolerance rate:

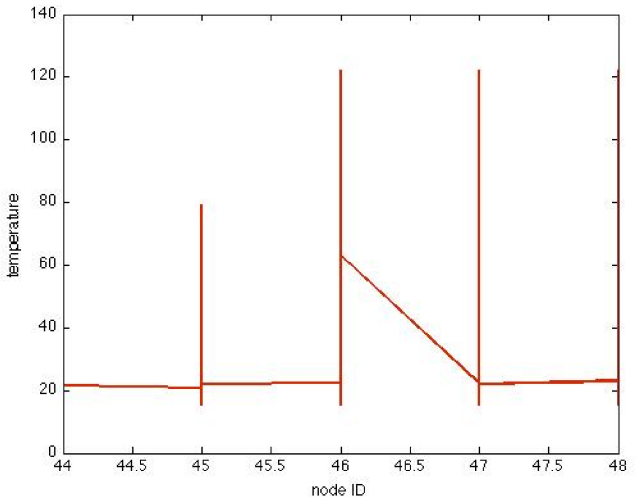
$$|\tilde{r_i} - r_{ij}| \leq \tau * \tilde{r_i}$$

Tolerance rate τ largely depends on characteristics and the intended application of the network. For most of the experiments we have set it to $\tau = 0.2$, or 20% margin.

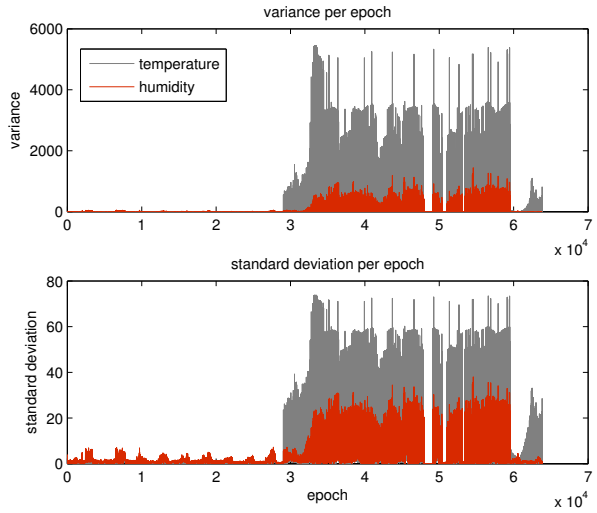
For the further analysis we use standard deviation $s^2 = \frac{1}{n-1}\sum_{i=1}^{n}(r_i - \tilde{r})^2$ and variance $\sigma = \sqrt{s^2}$

Figure 3 shows results of this analysis. We can see in Figure 3a which node has developed a fault, and in

 ISBN: 978-1-61208-207-3



(a) Node that has developed a fault



(b) Analysis to identify the time of possible fault development

Figure 3. Identifying nodes that have developed faults

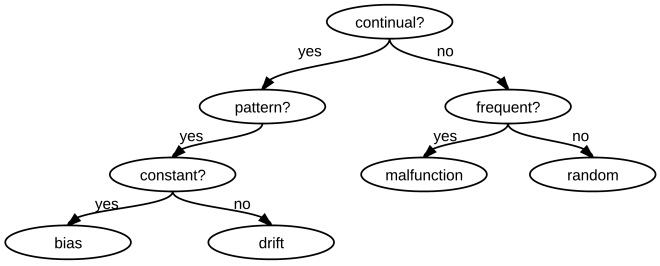


Figure 4. Decision tree

Figure 3b, we can see the time when the fault has started to develop. Also, here we can see that temperature and humidity measurements might be highly correlated. It might indicate that the node is becoming unreliable entirely.

Since we propose the use of a limited number of fault classes based on a small set of classification criteria, decision tree approach for pattern classification is a well fitting algorithm [16]. We can see this tree in the Figure 4. The outline of the process is given in the Algorithm 1.

Algorithm 1 Decision tree algorithm

Require: finite set of classes $C \in [\omega_1, \omega_2, \omega_3, \omega_4]$
  find the set of features $[f_1, f_2, ..., f_n]$
  for all data **do**
    analyze the time series
    check the continuity
    if ε_i is discrete **then**
      check the frequency
      if $\varepsilon_i > \tau$ **then**
        malfunction
      end if
      if $\varepsilon_i \leq \tau$ **then**
        random
      end if
    end if
    if ε_i is continuos **then**
      check the function $\varepsilon_i = f(t, [\alpha_1, \alpha_1....])$
      if $f(t, [\alpha_1, \alpha_1....]) = const$ **then**
        bias
      end if
      if $f(t, [\alpha_1, \alpha_1....]) \neq const$ **then**
        drift
      end if
    end if
  end for

Much of the work in designing trees focuses on on deciding which property test or query should be performed at each node. For this end, we have used MATLAB Treebagger function to run through various models of trees. This allows us to try many combinations and models of trees in a short time. Which combination will be suitable for a specific data set depends on many factors, and detailed discussion of this process is out of the scope of this paper.

Finally, in Figure 5, we can show at which point of time and which node has developed a drift with high probability.

At the current phase of the work, discovered fault is smoothed out. Since experiments are conducted on already collected dataset, there is no possibility to actually handle a faulty node in an adequate manner, and the comparison of the performance is left for the future work. Further development of the model requires experiments on simulated or deployed network, and it is also a part of future work.

Our experiments show around 15-20% occurrence of faults within the test data, which is consistent with other related experiments on the same dataset [7]. The method was able to correctly detect faults in approximately 90% of fault occurrences. These results are average from several

 ISBN: 978-1-61208-207-3

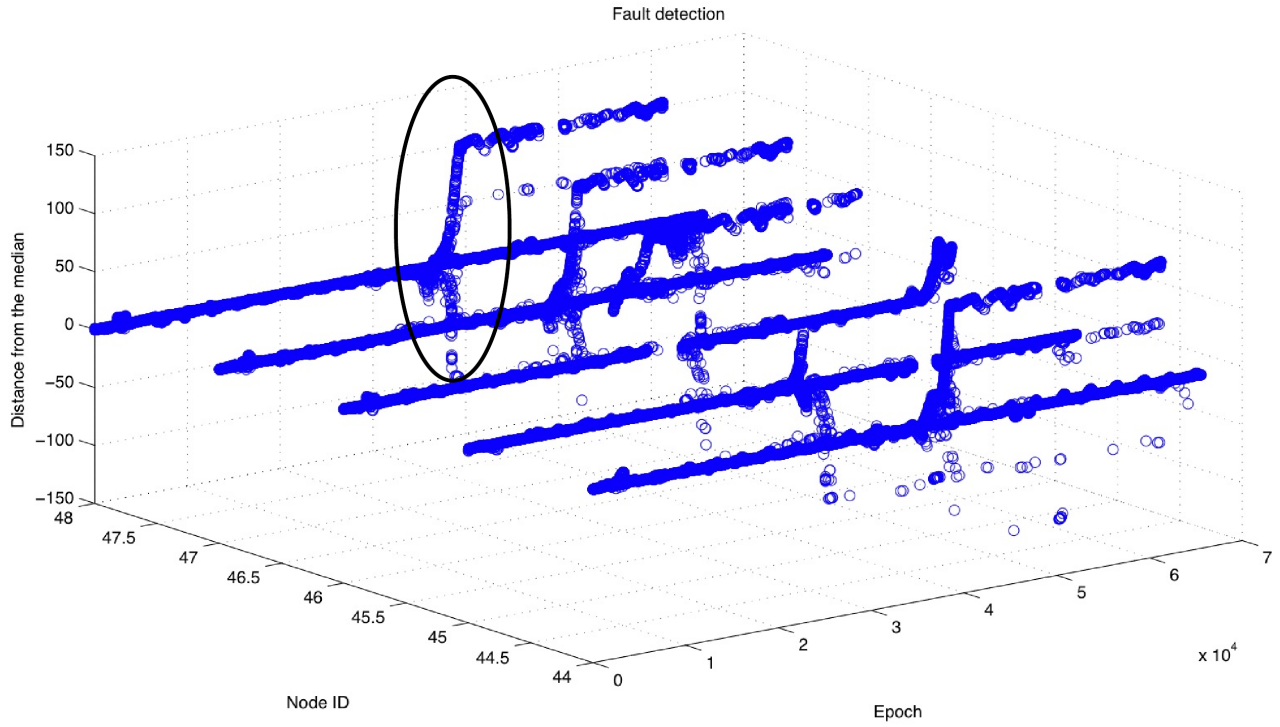


Figure 5. Drift development

runs on the same dataset. Minor differences occur due to the nature of Treebagger function which splits data into training and test set differently in each run.

At the moment, the method focuses on detecting the manifestation of faults in data, without looking into the underlying cause. However, a proper classification of a fault might indicate that cause. For example, malfunction is probably caused by an error in hardware, while drift might be caused by low battery levels or miscalibration. This provides for a flexible way of dealing with each node in an appropriate manner.

IV. Conclusion and Future Work

In this work, we focused efforts on discovering faults in sensor readings within a data-centric dense wireless sensor network. The focus is on the pattern of the occurrences of the faults on the node, regardless of the underlying cause. Key concept is a **classification of faults** based on **frequency, continuity and observable pattern** in the faults. We keep the assumption of spatial and temporal correlation between sensor readings. Also, we use centralized approach in order to ensure computational power with low communication overhead. In this way, the network is aware of when and where faults have developed, and it can handle a faulty node accordingly.

Statistical pattern recognition using decision tree method and classification work comparatively well for the data-centric and dense networks. However, if we can not assume density or spatial and temporal correlation of readings, this method is insufficient. It requires further development of a priori fault modeling to provide a basis for comparison of correct readings against faulty readings.

Providing an economic way of handling faults at the level of a single sensor is difficult. We believe that proposed classification, together with the detection method, provides a way to combine benefits of the centralized approach, such as bigger computational power with the flexibility of on-node correction. The network would be able to recognize fault on each node, handle it appropriately and keep the communication overhead low.

We have conducted experiments on Intel Berkeley dataset. So far they confirm that the method has a potential for flexible handling of faults in WSN. However, we plan to expand the experiments to use more datasets from different settings (urban and outdoor deployments) in order to test the hypothesis about spatial and temporal correlation.

Next, we plan to focus on a priori fault modeling and inclusion of developed models in the detection and classification phase. This would allow lesser dependence on neighboring nodes and stable connectivity.

To ensure the continually reliable operation, WSN has to be capable to address the full cycle of fault tolerance, which includes recognition of a faulty node, elimination or counteracting of its readings and self calibration where applicable. To meet this goal, we plan to expand proposed

 ISBN: 978-1-61208-207-3

method by introducing hypothesis finding techniques to discover fault models from the data. This is especially interesting for the continuos errors, such as drift or bias. If the network can learn these models, it can use them for correction of faults. It would enable the network to adapt its behavior and structure based on experience.

Finally, we would like to address some common issues that are often encountered when developing algorithms for WSN in the real world [17]. These issues include the assumption of a reliable communication, precise analysis of energy consumption in terms of communication and computation, synchronicity of readings and the assumption of a stable set of neighbors for each sensor node.

We believe that proposed method is viable for use in dense data-centric network setting with assumed good knowledge of network's topology. More importantly it provides a basis for further improvements and development of a complete scheme for self-repairing wireless sensor networks, which is the ultimate goal of the ongoing research.

ACKNOWLEDGMENT

The authors would like to thank A. Klein, F. Wagner, S. Toriumi, Y. Baba and F. Ishikawa, all of National Institute of Informatics, Japan, for their invaluable feedback. We would like to extend a special thanks for extensive discussion and support to G. Bourgne, also of National Institute of Informatics at the time.

REFERENCES

[1] F. Koushanfar, M. Potkonjak, and A. Sangiovanni-vincentelli, "Fault tolerance in wireless sensor networks, " book chapter," in *in Handbook of Sensor Networks, I. Mahgoub and M. Ilyas*, 2004.

[2] H. Karl and A. Willig, *Protocols and Architectures for Wireless Sensor Networks*. John Wiley & Sons, 2007.

[3] A. Hac, *Wireless Sensor Network Designs*. John Wiley & Sons, 2004.

[4] J. Shell, S. Coupland, and E. Goodyer, "Fuzzy data fusion for fault detection in wireless sensor networks," in *Computational Intelligence (UKCI), 2010 UK Workshop on*, September 2010, pp. 1 –6.

[5] M. Takruri, S. Challa, and R. Yunis, "Data fusion techniques for auto calibration in wireless sensor networks," in *Information Fusion, 2009. FUSION '09. 12th International Conference on*, July 2009, pp. 132 –139.

[6] K. Ni, N. Ramanathan, M. N. H. Chehade, L. Balzano, S. Nair, S. Zahedi, E. Kohler, G. Pottie, M. Hansen, and M. Srivastava, "Sensor network data fault types," *ACM Trans. Sen. Netw.*, vol. 5, no. 3, pp. 25:1–25:29, June 2009.

[7] A. B. Sharma, L. Golubchik, and R. Govindan, "Sensor faults: Detection methods and prevalence in real-world datasets," *ACM Transactions on Sensor Networks*, vol. 6, no. 3, pp. 1–39, June 2010.

[8] Y. Yao, A. Sharma, L. Golubchik, and R. Govindan, "Online anomaly detection for sensor systems: A simple and efficient approach," *Perform. Eval.*, vol. 67, pp. 1059–1075, November 2010.

[9] R. Tiwari and M. T. Thai, "On enhancing fault tolerance of virtual backbone in a wireless sensor network with unidirectional links," in *Sensors: Theory, Algorithms, and Applications*, ser. Springer Optimization and Its Applications, V. L. Boginski, C. W. Commander, P. M. Pardalos, and Y. Ye, Eds. Springer New York, 2012, vol. 61, pp. 3–18.

[10] G. Huang, Y. Zhang, J. He, and J. Cao, "Fault tolerance in data gathering wireless sensor networks," *The Computer Journal*, vol. 54, no. 6, pp. 976–987, 2011.

[11] R. Alena, R. Gilstrap, J. Baldwin, T. Stone, and P. Wilson, "Fault tolerance in zigbee wireless sensor networks," in *Proceedings of the 2011 IEEE Aerospace Conference*, ser. AERO '11. Washington, DC, USA: IEEE Computer Society, 2011, pp. 1–15.

[12] S. J. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach*, 2nd ed. Pearson Education, 2003.

[13] S. Marsland, *Machine Learning: An Algorithmic Perspective (Chapman & Hall/Crc Machine Learning & Pattern Recognition)*, 1st ed. Chapman and Hall/CRC, April 2009.

[14] F. C. Choo, P. V. Seshadri, and M. C. Chan, "Application-Aware Disruption Tolerant Network," in *Mobile Adhoc and Sensor Systems (MASS), 2011 IEEE 8th International Conference on*. IEEE, October 2011, pp. 1–6.

[15] I. B. Laboratory. (2012, June) Intel berkely dataset. [Online]. Available: http://db.csail.mit.edu/labdata/labdata.htmlf

[16] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification (2nd Edition)*, 2nd ed. Wiley-Interscience, November 2001.

[17] G. Halkes and K. Langendoen, "Practical considerations for wireless sensor network algorithms," *Wireless Sensor Network*, vol. 2, no. 6, pp. 441–446, June 2010.

 ISBN: 978-1-61208-207-3