Sensor Failure Detection and Recovery Mechanism Based on Support Vector and Genetic Algorithm

Jiehui Zhu[①], Yang Yang[①], Xuesong Qiu[①], Zhipeng Gao[①]

[①]State Key Laboratory of Networking and Switching Technology
Beijing University of Posts and Telecommunications
Beijing, China
E-mail:{jiehuizhu, yyang, xsqiu, gaozhipeng}@bupt.edu.cn

Abstract—**The main role of wireless sensor networks is to collect environmental data. As the sensor nodes are vulnerable and work in unpredictable environments, sensors are possible to fail and return unexpected response. Therefore, fault detection and recovery are important in wireless sensor networks. In this paper, we propose a fault detection algorithm based on support vector regression, which predicts the measurements of sensor nodes by using historical data. Credit levels of sensor nodes will be determined by a contrast between predictions and actual measured values. In this paper we also propose a fault recovery algorithm according to the node credit levels combined with genetic algorithm. The simulation results demonstrate that the algorithms we propose work well in failure detection rate, fault recovery speed and energy consumption.**

Keywords- fault detection; support vector regression; credibility level; fault recovery; genetic algorithm

I. INTRODUCTION

In recent years, with the development of microelectronic processing and battery technology, wireless sensor networks, with the advantages of rapid deployment and self–organization, achieve wide applications in environment monitoring. However, wireless sensors have to solve the problems of low reliability, limited battery energy and narrow transmission range.

As sensors suffer from the natural aging of electronic components inside and work in environments with numerous security threats, sensor faults exist inevitably. There is a kind of fault called software fault including the following situations: accuracy degradation, zero drift, data deviation and so on. Sensor nodes with software faults will collect wrong data. The area monitored by these failure sensors will become a leak and potential threat of the network. That will decrease reliability and quality of service of the wireless sensor network.

So far, some existing algorithms are inefficient enough. As the computing ability and energy of sensors are limited, these algorithms will be restricted. Also, it is unnecessary to replace all failure nodes under most circumstances. This paper presents a fault detection algorithm, making full use of the historical data so as to save energy and thus prolong the lifetime of sensor network. A heuristic fault recovery algorithm is also proposed in this paper. Through reducing the number of substitutive sensors and increasing the multiplexing of the routing path, our fault recovery algorithm manages to enhance the lifetime of wireless sensor network, as well as reduce the cost to replace sensor nodes.

The rest of this paper is organized as follows: Section 2 provides a brief review of related work. In section 3, we provide a detailed description of our fault detection algorithm. And in section 4, we provide a detailed description of our fault recovery algorithm. The performance evaluation can be found in Section 5. Finally, section 6 concludes the paper.

II. RELATED WORK

There are two kinds of fault detection algorithm, centralized and distributed. [1] describes Sympathy Centralized Algorithm. The algorithm requires that all nodes periodically exchange neighbor list, which will bring much energy consumption.

For large-scale WSNs, distributed algorithms are more appropriate in most cases. In [2], he node compares measured values with those of its neighbor nodes and if the absolute value of the difference is greater than a threshold, the node will be considered failure. [3] describes a fault detection algorithm based on clustering. The algorithm selects some good nodes as reference nodes, and then the other nodes are compared with the reference nodes to determine whether fault exists.

Venkataraman algorithm [4], proposed a failure detection and recovery mechanism towards energy exhaustion. Failure nodes below energy exhaustion threshold will notify its neighbor nodes before it completely shuts down. The nodes in the cluster are classified into four types. They proposed four different fault recovery mechanisms towards different types of nodes. Besides, G. Gupta and M. Younis [5] propose a method to recover from a gateway fault. When a gateway node fails, the cluster is dissolved and all member nodes reallocate to other gateways selected according to a backup list. An algorithm named Coverage Fidelity maintenance algorithm (Co-Fi) [6] is proposed to repair coverage loss based on sensor mobility. Co-Fi has the disadvantage of redundant energy consumption.

III. FAULT DETECTION ALGORITHM

Many fault detection algorithms work in all sensors in the same frequency. However, for most sensors with good fault detection history, the probability to fail is low. These sensors are not necessary to detect as often as those bad performance sensors. We propose Support Vector Regression Based Fault Detection Algorithm (SVR-based Fault Detection Algorithm, SFDA). SFDA has three steps as follow:

1. Training. Collect data sample string $T = \{(x_1, y_1), (x_2, y_2), \cdots, (x_L, y_L)\}$ with length L, L is 200~300. Then create the predictive array and train the parameters of support vector model.
2. Regression prediction. Use the Support Vector Regression Prediction to predict the data at t_{i+1} by data already collected , which is $y_{i+1} = f(x_i)$.
3. Dividing credit levels. Compare the prediction with actual value $d_m = x_{i+1}$, calculate credit value λ_{i+1}. Then divide credit levels according to λ_{i+1}.

In the process of classifying credit levels, Support Vector Regression (SVR) Prediction Algorithm is the core procedure of SFDA. The authority of credit level classification depends on the accuracy of the prediction.

Based on Statistical Learning Theory (SLT), Vapnik and his colleagues proposed a Support Vector Machine (SVM) learning theory [7]. SVM is based on Principle of Structural Risk Minimization. SVR, a data-oriented machine learning algorithm, is the crucial application of SVM. In virtue of

preliminary effective data, SVR constructs a training set, and defines the internal correlation to predict the future data.

We assume that, after normalizing, actual sensor data and the predicted data towards it compose of a sample set. There is a certain probability distribution $F(x, y)$ in the set. We select independent and identically distributed sample points in the set and analyze them. However, as the prediction is hardly completely the same as the actual data, loss function is used to measure structural risk in SVM theory. We use $Loss(y, f(x))$ to represent loss function, and calculate the mathematical expectation of structural risk by formula (1) below.

$$R[f] = \int Loss(y, f(x)) dF(x, y) \quad (1)$$

According to the principle of Structure Risk Minimization (SRM), in order to achieve the best prediction effect, we need to find out the optimal prediction function $f(x)$ to minimize $R[f]$. $R[f]$ also expresses the distance between any point in the sample set and the hyperplane. A hyperplane in the multi-dimensional space can be represented as:

$$f(x) = \langle \omega, x \rangle + b \quad (2)$$

On the basis of the principle of Structure Risk Minimization, getting the minimal value of $R[f]$ is equal to adjust the gradient ω to maximize the distance between the nearest point and the hyperplane, which means maximizing $\varepsilon / \sqrt{1 + \|\omega\|^2}$. The division of this hyperplane with this gradient ω has the best prediction results. We called it the optimal hyperplane. In this paper we use the ε- insensitive loss function, the solution of hyperplane to achieve structural risk minimization is:

$$Min \quad \frac{1}{2}\|\omega\|^2 \quad (3)$$

$$s.t. \quad |(\langle \omega, x_i \rangle + b) - y_i| \leq \varepsilon, \ \ i = 1,2, \cdots, L \quad (4)$$

To solve the problem, we introduce Lagrange Multipliers $\alpha = (\alpha_1, \alpha_2, \alpha_3, \ldots, \alpha_L)$ and turn the problem above into its dual programming problem, and then cited KKT conditions. Then We can get the solution:

$$f(x) = \sum_{i=1}^{L} (\alpha_i - \alpha_i^*) \varphi\langle x_i, x_j \rangle + b \quad (5)$$

Above, $\varphi\langle x_i, x_j \rangle$ is an inner product between two mapping in high-dimensional feature space and difficult to solve out. That is a "Curse of Dimensionality". In the SVM theory, the inner product computation is solved by kernel technique. Kernel technique tries to find a kernel function, qualified $K(x_i, x_j) = \langle x_i, x_j \rangle = \varphi(x_i) \bullet \varphi(x_j)$, aiming to avoid the cost of inner product computation in high-dimensional space by kernel function substitution. In this paper, we choose polynomial kernel function, $K(x_i, x_j) = (x_i \cdot x_j + 1)^d$, d=1,2,…,N, which is the most mature. In polynomial kernel function, d is the power which defined according to the WSN size. Substituting into (5), we can get the prediction function of WSN sensor data:

$$f(x) = \sum_{i=1}^{L} (\alpha_i - \alpha_i^*) (x_i \cdot x_j + 1)^d + b \quad (6)$$

We can define the credit value λ_i as the residuals of the actual value and the prediction results calculated by SVR :

$$\lambda_i = \frac{|f(x) - y|}{y} \quad (7)$$

Based on credit values λ_i, we divide sensor nodes into five credit levels: SURE, GOOD, COMMON, DOUBT and FAULT. Different credit levels lead to different action towards the sensor fault counter. The sensor fault counter initials to be zero. If the counter is larger than the fault threshold F_{th}, the sensor is considered to be failure and send a fault report to the sink node. Sensors with high credit level have low fault detection frequency to save energy. In addition, Sensor node maintains a table which combined routing paths with credit levels. Sensor nodes select neighbor with highest credit level when routing towards the sink node. The evaluation standard and detailed actions of each credit level are as Table 1.

Table 1: the credit levels and detailed actions

Credit Levels	Division Standard	Detailed Action
SURE	credit value<5%	Reduce fault detection frequency and clear the fault counter
GOOD	5%<credit value<=10%	Reduce fault detection frequency and the fault counter decrease by 1
COMMON	10%<credit value<=30%	Remain fault detection frequency unchanged
DOUBT	30%<credit value<=50%	Raise fault detection frequency and the fault counter increase by 1
FAULT	50%<credit value	Ensure itself to be a fault node, and send the fault report

IV. FAULT RECOVERY ALGORITHM

This paper proposes a fault recovery algorithm named Credit Levels-based Genetic Algorithm (CL-GA) for WSNs based on credit levels calculated by SFDA combined with genetic algorithm. During the fault detection stage, the SFDA algorithm calculates the credit levels, neighbor list and node degree of each sensor node. Then each failure node calculates N_{th} according to (8). N_{th} is used to measure the ability of neighbors to collect data instead of the failure sensor. If N_{th} is less than the threshold ∂, the CL-GA will be invoked. CL-GA computes the IDs of sensors which should be replaced based on genetic algorithm to minimize the sensors to be replaced and maximum the usage of routing paths. Then the wireless sensors adjust the routing paths and credit levels and continue to work.

$$N_{th} = \frac{\sum_i \omega_i}{d_i} \quad (8)$$

$$\omega_i = \begin{cases} 1 & \text{If node } i \text{ is SURE or GOOD or COMMON} \\ -1 & \text{If node } i \text{ is DOUBT or FAULT} \end{cases} \quad (9)$$

In (8), d_i means the node degree of sensor i, defining the number of neighbor nodes.

In order to recover the WSN rapidly and minimize the recovery cost, we take the partial replacement strategy. Only part of failure nodes will be replaced. Genetic algorithm, which is used inCL-GA, is a heuristic search algorithm referring to biological evolution law to solve optimization problems in artificial intelligence.

In CL-GA, parameters are encoded in binary characters and serve as the chromosomes. Fitness value describes the extent of each individual to adapt to the environment. In CL-GA, genes on the chromosome will be adjusted to maximize the fitness value in order to approach the optimal solution. A fitness function is introduced to generate the fitness value for each chromosomes. Adjusting genes on the chromosome to achieve extreme fitness value is equivalent to calculating the extremum of fitness function. CL-GA consists of five steps: initialization, evaluation, selection, crossover and mutation as follows:

A. Initialization

In the initialization step, the CL-GA generates a series of chromosomes, and each chromosome is an expected solution. The length of the chromosome represents the number of the failure node. Here is an example of chromosomes in Figure 1. In each gene (bit), a '1' means the node should be replaced, and a '0' means that the node should stay. These chromosomes compose of the initial population P(0). The quantity of the chromosomes which is defined by the user is on behalf of the population size, and the initial population P(0) represents the set of initial feasible solution of CL-GA. During the

initialization, we set an evolution generation counter t=0. Besides, we also set the maximum evolution generation T which is stipulated by user. The higher T is set, the better result of the genetic algorithm, while expending more time.

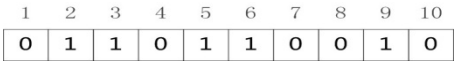

Figure 1.Chromosome and its gene

B. Evaluation

In the evaluation step, the algorithm calculates the fitness value of each individual of population P(t) according to the fitness function. Generally, fitness value is calculated by the fitness function, and the parameters of the fitness function are the genes on chromosome. However, in CL-GA, genes can't be applied directly to the fitness function, because the genes on chromosome only indicates whether the sensor to be replaced. In the fault recovery algorithm, the genetic algorithm is also used to maximize the reuse of the routing path as well as to achieve the purpose of recovering the fault with least nodes to be replaced. Therefore, considering the above two optimization goals, we can work out the following fitness function:

$$fitvalue = (\frac{N_{rec}}{N_{all}})^{-1} \times \sum_{i \in nei} path(i) \qquad (10)$$

In (13):

N_{rec} =the number of fault nodes to be replaced when recovery.

N_{all}= the whole number of fault nodes.

path(i)= effective degree of the neighbor node *i*. We get each node *i* from the neighbor list.

p_{12}, p_3, p_4= effective degree when the neighbor node is SURE or GOOD/ COMMON/ DOUBT. Usually, p_{12} is 1. p_3 and p_4 $are\ less\ than$ 1.

C. Selection

In the selection step, we expect the individuals with relatively lower fitness values to be removed, while the remaining individuals with higher fitness values will be retained. We adopt the principle of elitism. The top 50% will be retained and put in the mating pool, while the remaining 50% will be deleted. In CL-GA, we use "roulette wheel selection" to select the optimal individuals. Chromosomes with better performance in fitness value will have higher probability to retain. Poor chromosomes will be deleted and replaced by new chromosomes generated in the crossover step.

D. Crossover

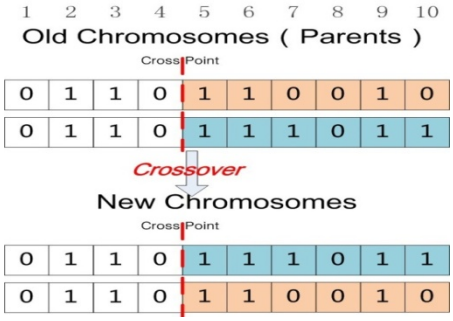

Figure 3.Crossover step

In the crossover step, we change the chromosome in mating pool by one-point crossover. One-point crossover will choose a bit and exchange parent individuals' chromosomes fraction after it. Two individuals are selected to be parents while a crossover point is selected randomly between the first and last genes of the parent individuals. Then, the fractions after crossover point are exchanged and concatenated to create new chromosomes. Crossover step is as Figure 3.

E. Mutation

In the mutation step, we introduce several binary mutations. There are two reasons: first, to ensure the local random search capabilities; second, to maintain the diversity of population to prevent premature convergence. In CL-GA, each chromosome has a probability p_m to mutate. Once selected, a gene of it will be chosen randomly and flip (0->1 or 1->0). Mutation step is as Figure 4.

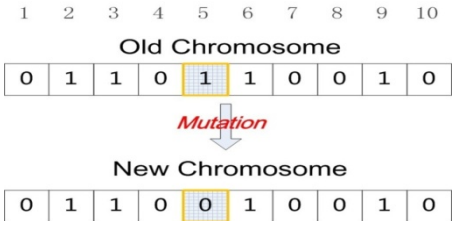

Figure 4.Mutation step

V. PERFORMANCE EVALUATION

In this section we evaluate the performance of our proposed algorithms. In our simulation, we assume that all nodes are the same and static. What's more, wireless sensor node has two transmission modes. Nodes are usually in the normal mode with the transmission radius of 30 meters. If there are no neighbor nodes for routing, the node will turn to the high-power mode. In high-power mode, the transmission radius will increase significantly, while the energy consumption will greatly increase at the same time. The simulation parameters are explained in Table 2.

Table 2: Simulation parameters

Parameter	Description	Value
Coverage area	The size of simulation area	500×500m
Node Quantity	Number of nodes	100, 150, 200, 250
Coverage radius	Coverage radius in regular mode	50 m
Energy	Initial energy available in each node	1000 mJ
Distribution	Node distribution in WSN	Random
Packet loss rate	Probability of losing a packet in transmission	0 bps

A. Fault Detection Algorithm

We compared SFDA with the algorithm in literature [2] and literature [3] in order to verify the advantage fault detection rate. In literature [2], we promote the Localized Event Boundary Detection Algorithm (LEBDA), which is a local algorithm to determine whether there is a failure sensor on the basis of the local neighborhood data. While in literature [3], the Tendency-based Localized Faulty Sensor Detection Algorithm (TLDA) is based on clustering. In this algorithm, the cluster head node will choose credible reference nodes from cluster members, so we can detect node faults according to the data of the reference nodes. The simulation will run in different network scales (100,150,200,250 nodes respectively) with each network scale repeated 50 times. The simulation result of fault detection rate is shown in Figure 5.

It can be obtained from Figure 5 that SFDA algorithm is much better than the contrastive algorithm in fault detection rate. The fault detection rate of SFDA remain above 94% in different network scales, which is higher than the result in LEBDA of about 87% and that in TLDA of about 84%. In addition, in LEBDA and TLDA, the fault detection rate shows a downward trend when the network scale increases. However, the failure detection rate of SFDA algorithm proposed in this paper remains stable when the network scale increases. From the above analysis, we obtain that SFDA algorithm can reach a

considerable fault detection rate. Besides, the algorithm results will not be affected when the network size increases.

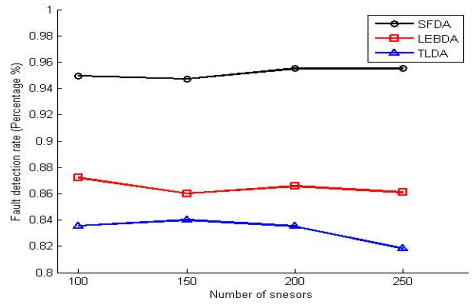

Figure 5. Comparison of fault detection rate

B. *Fault Recovery Algorithm*

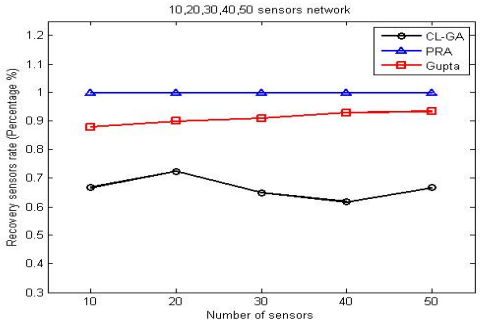

Figure 8(a). Amount of recovery sensors for small networks

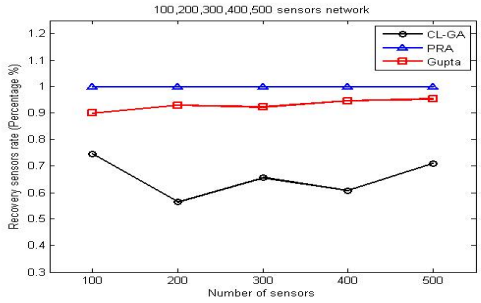

Figure 8(b). Amount of recovery sensors for large networks

When a wireless sensor network has a leak, we expect to recover the network as soon as possible by replacing reasonable amount of failure nodes. It is usually unpractical and over-demand to replace all failure nodes. In this part, we compare the number of recovery nodes of CL-GA with two advanced algorithms: periodical replacement algorithm (PRA) and Gupta algorithm [5]. The parameters of CL-GA are explained in table 3. And PRA records the positions of failure nodes and replaces all of them when the network coverage is too small. In Gupta algorithm, when a sensor node fails, the routing responsibility delegates to the neighbor node with most energy. When a sink node fails, the cluster will be dissolved. Cluster members reallocate to the pre-selected sink node. The result is shown in figure 8(a) and figure 8(b).

As the core mechanism of CL-GA is a heuristic algorithm, the result of CL-GA is unstable, especially when the WSN is large, as figure 8(b). However, CL-GA does well in reducing the recovery sensors with only 60%~70% sensors to be replaced. In comparison, periodical replacement algorithm replaces all fault nodes. It leads intolerable data loss and waste. Gupta algorithm serves a little better. As Gupta algorithm solves the sink node faults by reallocation tactics, it only replaces all of the non-sink nodes. But Gupta algorithm is still not inefficient enough (still over 90%). In summary, CL-GA replaces the least nodes in fault recovery so as to resume the WSN quickly and economically.

Table 3: CL-GA parameters

Parameter	**Value**
Population Size	20
Chromosome Length	The same as the number of fault nodes
Recovery threshold Nth	3.0
Probability of Crossover	0.6
Probability of Mutation	0.001
Maximum evolution generation	20
Initial Population	Random

VI. CONCLUSION

Because of the limited resource, low reliability and dynamic network change, wireless sensor networks are confronted with sensor faults. We propose SFDA which combines support vector regression prediction with credit levels division to reduce the energy and computation consumption and increase the fault detection rate. Besides, reusing the credit levels in fault detection, we propose CL-GA. Through initialization, evaluation, selection, crossover and mutation, CL-GA computes the optimal solution to recover the WSN with least replaced sensors and most re-usage of routing path.

Through the simulation of the real environment, we conclude that SFDA performs well in fault detection rate. CL-GA, a further application for credit levels calculated by SFDA, does well in rapid recovery. In the future research, we will study how to optimize our algorithms for more stability.

ACKNOWLEDGMENT

This work was partly supported by Funds for Creative Research Groups of China (61121061), NSFC (61272515, 61372108), Ph.D Programs Foundation of Ministry of Education of China (No. 20110005110011), Fundamental Research Funds for the Central Universities (No. 2014RC1102), and Beijing Higher Education Young Elite Teacher Project (YETP0474) .

REFERENCES

[1]. Ramanathan N, Chang K, Kapur R, Girod L, Kohler E, Estrin D. “Sympathy for the sensor network debugger[J]” Proceedings of the 3rd international conference on Embedded networked sensor systems, Nov 2005: pp:255-267.

[2]. Ding M, Chen D, Xing K, Cheng X. “Localized fault-tolerant event boundary detection in sensor networks[C]” INFOCOM 2005. 24th Annual Joint Conference of the IEEE Computer and Communications Societies. 2005: pp: 902-913.

[3]. Chen J, Kher S, Somani A. “Distributed fault detection of wireless sensor networks[C]” Proceedings of the 2006 workshop on Dependability issues in wireless ad hoc networks and sensor networks. 2006: pp: 65-72.

[4]. G. Venkataraman, S. Emmanuel, S. Thambipillai, “Energy-efficient cluster-based scheme for failure management in sensor networks[C]” IET Common, Volume 2, Issue 4, April 2008: pp: 528-537.

[5]. G. Gupta and M. Younis, “Fault tolerant clustering of wireless sensor network[C]”, WCNC’ 2003: pp: 1579-1584

[6]. S. Ganeriwal, A. Kansal, M.B. Srivastava, “Self aware actuation for fault repair in sensor networks[C]”, IEEE International Conference on Robotics and Automation (ICRA), May 2004: pp: 5244-5249

[7]. C Cortes and V Vapnik. “Support-Vector Networks[J]” Machine Learning, 1995 ,20: pp :273-397