A Faulty Node Detection Method in Wireless Sensor Network in Seedling for Hydroponics

Dong-Hee Noh, Tae-Hwan Ko, Ahhyeon Hong, Kyeong-Hun Kim, Seok-Bong Noh*
*IT Application Research Center, Jenbuk Regional Branch, Korea Electronics Technology Institute (KETI), Jeonju, Republic of Korea
{dhee.noh, kth1147, ahhyeon0501, khkim, mogo142}@keti.re.kr

***Abstract*—This paper proposes a faulty node detection scheme in the hydroponics system. A wireless sensor system of seedling hydroponic has to considered distance-dependent bandwidth, poor quality of the links, and propagation delay. In the proposed scheme, the main control sensor node can diagnosis which sensor node are error frequently occurs. Simulation results show that the proposed method can improve the performance of the faulty detection reliability and miss-detection rate for real-time hydroponics wireless sensor systems.**

***Index Terms*—Greenhouse, Hydroponics, Faulty Node Detection, Faulty Weighting Factors, BCH.**

I. Introduction

The robustness of wireless communication systems has been investigated for smart-farm application [1]. Especially, the design of underwater networks of hydroponics system, which is significantly affected by the limited and distance-dependent bandwidth, poor quality of the links, and propagation delay (low speed of sound), which differentiate underwater communication from terrestrial wireless networks.

The recent research trend is focused on improving the performance of real-time communication for the aquaponics system, with the desired outcome being communication performance. Many studies also have been conducted to develop networking solutions for underwater sensor networks, including acoustic channel modeling and physical layer transmission analysis as well as networking protocols [2].

This paper investigates the possibility more reliable faulty node detection scheme in distributed systems with the use of the Markov chain. The Markov-chain model is suitable for distributed systems, which depend on probability computation for solving recent network problems such as throughput, redundancy, and packet retransmission [3].

In this paper, the Markov chain is applied in decision judgment where the probability between **Normal** and **Standby-state** is calculated based on independence in master and slave nodes. The master and each slave node compare the single BCH codes to detect the faulty node. One fundamental scheme is associated with reliability performance through the Markov chain computation, which depends on the probability. Therefore, the main contribution is to enhance the reliable decision of faulty node detections in the decoding process.

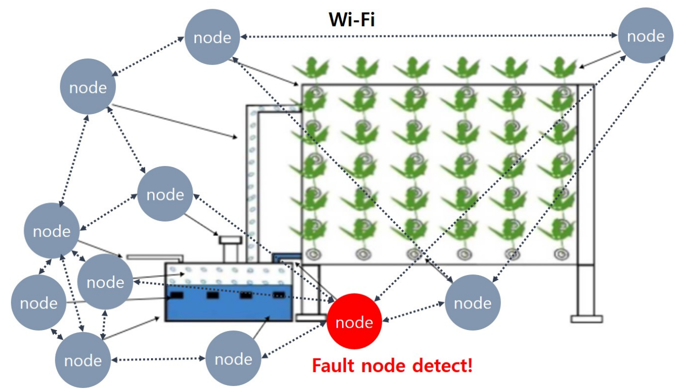

Fig. 1: The Basic Concept of the Hydroponics Seedling Cultivation with Wireless Sensor Network System.

II. Proposed scheme

A. Initialization Step

To compute the discrete-time Markov chain, we assume that **P** is the transition probability matrix, as defined by equation [1].

$$\mathbf{P} = \begin{pmatrix} p_{II} & p_{IJ} \\ p_{JI} & p_{JJ} \end{pmatrix} \quad (1)$$

where I and J are different elements.

We consider the set **S** for three states through the Markov chain model. In this paper, we assume that $\mathbf{S} = \{1, 2, 3\}$, where the first state denotes the **Normal-state**, the second state denotes the **Standby-state** and the third state denotes the **Fault-state** of this system.

First, we consider the recurrence state. In this state, the I state satisfies this condition : $I \subseteq S$, which can denotes the recurrent state when the transition direction is the same, i.e. P_{II}, P_{JJ} or P_{KK}. Next, we denote R_I, as the expected value associated with the probability P_I. We also denote τ_I as the return time to state I.

$$\tau_I = P\{A|X_0 = I\}, E_I = \sum_{n=0}^{\infty} I_i, \quad (2)$$

B. Transition Step

When we computing the k-step Markov chain model, the recurrent state matrix depends on the following two probabilities: the stationary probability P_s and the threshold Probability P_t.

Using these parameters, equation [2], we can define the equation [3]. for the recurrent state matrix,

$$\begin{aligned} P_{II} &= RE_I P_{Tansient} \\ &= \sum_{k=1}^{\infty} k P_t^{k-1}(1-p_t)(\tau_i P_s p_t). \end{aligned} \tag{3}$$

where, $P_{Tansient}$ is the probability of recurrence at **equal-state**.

We define the transition probability from state I to state J as P_{IJ}. We assume that the system uses the discrete time Markov chain model with general finite state space S, that is restricted to recurrent state. Before describing how to calculate the P_{IJ}, we define several parameters. First, we denote α_k as the initial condition state matrix. Normally, p_g and p_b are included in α_k, i.e. (p_g, p_b). Next, we define the S^n as the stationary distribution for the original Markov chain model using equation [4]. We can combine S^n and P_s to makes k-step Markov chain model computation using equation [4].

$$S^n = \begin{cases} a_g \prod_{i=1}^{n-1} P^i & \text{if algorithm is } \textbf{Normal-state} \\ a_b \prod_{i=1}^{n-1} P^i & \text{if algorithm is } \textbf{Standby-state}, \end{cases} \tag{4}$$

$$\begin{aligned} P_{IJ} &= \alpha_k m_k P^c_{Tansient} \\ &= \prod_{j=1}^{m} \alpha_k^j \sum_{i=1}^{n} S^i P_s p_t P_{I-1J-1}, \end{aligned} \tag{5}$$

where, $P^c_{Tansient}$ is probability of recurrence at **different-state**.

We can obtain the transition probability through the Markov chain computation m_k and α_k together using equation [5].

III. Proposed Algorithm

A. Normal-State Mechanism

The basic idea of a **Normal-state** algorithm, which is that if the interval for computing the distance between two errors is less than a threshold, then the node state should be recalculated. If the error sequence from a certain node is not found at t_n, and the next error occurred at t_{n+1} where t_{n+1} - t_n is less than threshold time, then the node state will be changed based on the Markov chain. If the **Normal-state** should be changed from **Normal** to **Standby-state**. A similar approach is also used for decoding when a **Standby-state** should be changed to a **Fault-state**, in which case, the node is confirmed as a faulty node.

Initially, all of the nodes are in the **Normal-state**. For examining of each slave node, the system used the state-matrix : $\alpha_g = (a, b)$, where a is the **Normal-state** probability p_g and b is **Fault-state** probability p_b.

The system observes the time interval between two errors, ti_o, to check if it is within the maximum threshold probability, ti_{tg}. If ti_o is smaller than ti_{tg}, the master node recognizes that the system may have a faulty node problem in real-time. Based on this consideration, the system processes the Markov-chain computation through equation [6].

The initial state of the Markov-matrix is denoted as B, and consist of the elements of p_{gg}, p_{gb}, p_{bg} and p_{bb}. p_{gg} indicates the probability computation between **I-state** to **Normal-state**. p_{gb} indicates the probability computation between each **Fault-state**. p_{gg} indicates the reverse computation of p_{gb} and p_{bb} indicates the reverse computation of p_{gg}, p_{gb}, p_{bg} and p_{bb} are 0.7, 0.3, 0.2 and 0.8, respectively.

After computing the Markov model, the system checks the following condition. If the element of the state matrix A_{12} is greater than the probability limit for **Fault-state**, p_{tb}, the system moves to the **Standby-state**. Equation [8] describes the transition from **Normal** to **Standby-state**,

$$P_{gb} = \prod_{j=1}^{m} \alpha_g^j \sum_{i=1}^{n} S^i P_s p_t P_{01}. \tag{6}$$

where $p_{tb} > p_{tb}$.

On the other hand, if ti_o is less than ti_{tg}, the system normalizes the state-matrix. During normalization, A_{11} adds the 0.1 value and A_{12} subtracts the 0.1 value. It can is computed using equation [7],

$$P_{gg} = \sum_{k=1}^{\infty} k P_t^{k-1}(1-p_t)(\tau_i P_s p_t). \tag{7}$$

where t is the observation time and n is the total operation time in **Normal-state** algorithm.

B. Standby-State mechanism

After examining the state of the slave nodes if the elements of the state-matrix A_{21} is greater than the probability limit value for the **Fault-state**, p_{tb}, then the master node processes the **Standby-state** algorithm. In this case, all of the nodes are assumed to be initially in the **Fault-state**. For examining each slave node, the system makes the revised state matrix: $\alpha w = (C_{11}, C_{12})$ where C_{11} is the p_b and C_{12} is p_g, which is reversal of the original matrix, A. The master node checks the state of each slave node by using the reverse-state-matrix, αb.

During the process, the system detects the fault state in real-time. If the interval time between two errors occurs inside the time limit, the state matrix will be normalized by incrementing the **Fault-state**. On the contrary, if no errors occur inside the time limit, then the Markov chain operation is applied to the state matrix as described by equation [8].

$$P_w = \begin{cases} P_{wb} = \prod_{j=1}^{m} \alpha_w^j \sum_{i=1}^{n} S^i P_s p_t P_{gw} & \text{if } e_t < e_m, \\ m_k = P_s \sum_{i=1}^{n} S^i & \text{if } e_t \geq e_m. \end{cases} \tag{8}$$

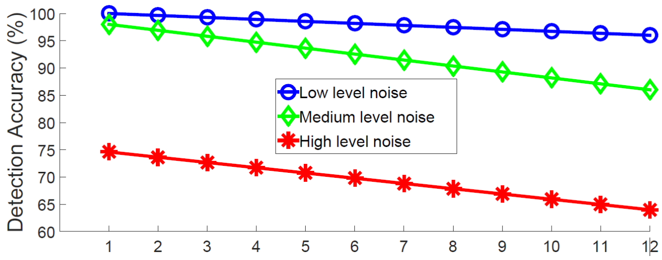

Fig. 2: Performance of Correlation between Detection Accuracy and the Total Number of Faulty Nodes.

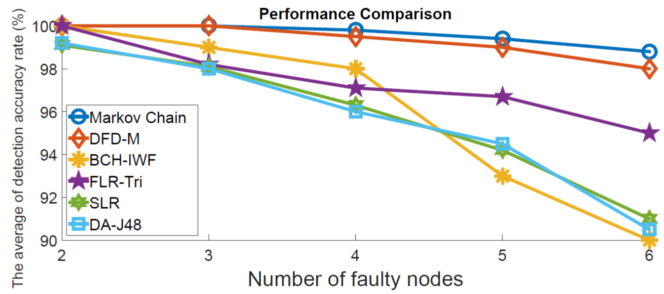

Fig. 3: Performance Comparison of Detection Accuracy Rate.

where k is the Markov chain computation.

After computing the state matrix, the system checks the value of the current state matrix. If the element of the state matrix C_{12} is greater than the probability limit for the **Normal-state**, p_{tg}, the flow algorithm goes to the **Normal-state**. Otherwise, the state retrain its **Standby-state** in real-time. equation [9] describes the following step.

$$P_w = \begin{cases} P_{wg} = \prod_{j=1}^{m} \alpha_w^j \sum_{i=1}^{n} S^i P_s p_t P_{go} & \text{if } p_g > p_{tg}, \\ P_{ww} = \sum_{k=1}^{\infty} k P_t^{k-1} (1-p_t)(\tau_i P_s p_t) & \text{if } p_g \leq p_{tg}. \end{cases} \quad (9)$$

where t is the observation time and l is the total operation time in the **Normal-state** algorithm.

IV. Simulation

Figure 2 shows the correlation between the detection accuracy and the total number of faulty nodes. The x-axis shows the number of faulty nodes inside the system, from 1 to 12 faulty nodes. The system is embedded with the proposed method. Three different levels of channel noise are then introduced into the system: low, medium, and high.

Similar results were obtained even when the total number of faulty nodes is increased to 12. Furthermore, the detection accuracy between 1 and 12 faulty nodes is similar even when the channel noise is quite high. The effect of the channel noise is stronger than that of the false signal caused by the number of faulty nodes. Therefore, the performance of the faulty node detection is nearly the same in an environment with a higher channel noise.

Tri, SLR [4], BCH insertion scheme with Interval Weighting Factor (BCHIWF) [5], DA-J48 [6], and the proposed method (Markov Chain). In this simulation, one node is set with faulty behavior with a star topology. Each method will try 1000 times to detect the faulty node. Figure 3 shows that the proposed method (Markov Chain) and DFD-M achieve almost 100% with two and three faulty nodes. Afterward, the proposed method is slightly superior to DFDM. Although BCH-IWF achieves 100% with one faulty node, when there are three and five faulty nodes, it is sharply declined. BCH-IWF even has the lowest accuracy with six faulty nodes.

The detection accuracy using FLR-Tri is 95.1% and only 91.3% in SLR with six faulty nodes. Moreover, DAJ48 has the lowest average. Figure 3 proves that the proposed method is superior to any recently investigated algorithms.

V. Conclusion

In this paper, the proposed scheme enhances the master node's capability to accurately predict the interval in which an error frequently occurs based on a Markov Chain model. This scheme improved the performance regarding the detection reliability and detection accuracy rate.

In future work, we will research the enhancement method of the real-time performance of a networked hydroponics system communicated with a continuously controlled agriculture automotive robot.

Acknowledgment

This work was supported by Korea Institute of Planning and Evaluation for Technology in Food, Agriculture and Forestry(IPET) through Smart Farm Innovation Technology Development Program Project, funded by Ministry of Agriculture, Food and Rural Affairs(MAFRA), Ministry of Science and ICT(MSIT) and Rural Development Administration(RDA) (421035-04-1-CG000)

References

[1] L. D. Xu, W. He, and S. Li, "Internet of Things in Industries: A Survey," IEEE Transactions on Industrial Informatics, vol. 10, no. 4, pp. 2233–2243, 2014

[2] A. Stefanov and M. Stojanovic, "Design and performance analysis of underwater acoustic networks," IEEE Journal on Selected Areas in Communications, vol. 29, no. 10, pp. 2012 –2021, Dec. 2011.

[3] S. X. Ding, S. Yin, K. Peng, H. Hao, and B. Shen, "A Novel Scheme for Key Performance Indicator Prediction and Diagnosis With Application to an Industrial Hot Strip Mill," IEEE Transactions on Industrial Informatics, vol. 9, no. 4, pp. 2239–2247, Nov 2013.

[4] Y. Yang, Q. Liu, Z. Gao, X. Qiu, and L. Meng, "data fault detection in medical sensor networks"," Sensors, vol. 15, no. 3, pp. 6066–6090, 2015.

[5] A. Prasetiadi and D. S. Kim, "Faulty Node Detection in Distributed Systems Using BCH Code," IEEE Communications Letters, vol. 17, no. 3, pp. 620–623, March 2013.

[6] L. B. Ruiz, I. G. Siqueira, L. B. e. Oliveira, H. C. Wong, J. M. S. Nogueira, and A. A. F. Loureiro, "Fault management in event-driven wireless sensor networks," in Proceedings of the 7th ACM International Symposium on Modeling, Analysis and Simulation of Wireless and Mobile Systems. New York, USA: ACM, 2004, pp. 149–156.