Reliability - Efficient Mechanism (AJIA) of IoTs for Packet Loss Recovery and Route Performance Evaluation

Faisal, falharbi5@toronet.csudh.edu
Jieli Chen, jchen100@toromail.csudh.edu
Mohammad, malthubyani1@toromail.csudh.edu
Department of Computer Science, California State University, Dominguez Hills

***Abstract*—The reliability of real time emergency applications for IoTs is one of the biggest challenges in wireless telecommunications since Fast response is required in critical situations (fire detection, terrorist attack, etc). Solutions on minimizing loss recovery and latency with higher connecting reliability of IoTs are critical in IoTs. This paper focuses on one of the solution to improve reliability of emergency applications under IoT technology, called Adaptive Joint protocol based on Implicit ACK (AJIA) mechanism for packet loss recovery and route quality evaluation in the IoT. In this mechanism, the overhearing feature is used, characterizing the wireless channels, as an implicit ACK mechanism. In addition, the protocol allows for an adaptive selection of the routing path based on the link quality. We evaluate performance with a simulator with two setup models: 1). meeting room or home environment with less than 10 sensing devices; 2). small working office or classroom environment with less than 25 sensing devices. The results show that around 80% of sensing devices links increase connecting reliability and the average increases are 11.2% and 5.7% respectively for two different models. For future work, we intend to evaluate the effectiveness of AJIA mechanism in terms of energy efficiency.**

I. INTRODUCTION

The Internet of things (IoTs) is the network of physical objects or "things" which are electronics, software, sensors, and network connectivity which enable these things collect and exchange data[1]. We can get this by gaining substantial ground in modern wireless telecommunications [2]. In general, we can understand IoTs is a mechanism where heterogeneous objects and the working of IoT scheme is based on the IP addressing. The goal is connecting a variety of things around us to the Internet. In fact, developers are seeking to make (IoTs) self-configured dynamic global network infrastructure. All of devices, the physical or the virtual devices, have attributes, personalities. All of these devices are seamlessly integrated into the information infrastructure. Additionally, the inclusion of the (IoTs) concept into the emergency application will open new perspectives by identifying the ‘things’, achieving sensing tasks and building low cost and reliable solutions and services [3]. Also, fast response is required in critical situations (fire detection, terrorist attack, etc). Because of that, using Adaptive Joint protocol based on Implicit ACK (JAIA) Mechanism will help to improve the reliability of Internet of Things and making the reliability more useful.

II. RELATED WORK

Industrial Wireless Sensor-Actuator Networks (WSANs) enable Internet of Things (IoT) to be incorporated in industrial plants. The dynamics of industrial environments and stringent reliability requirements necessitate high degrees of fault tolerance. WirelessHART employs graph routing to enhance network reliability through multiple paths [4]. Since many industrial devices operate on batteries in harsh environments where changing batteries is prohibitively labor-intensive,

WirelessHART networks need to achieve a long network lifetime. The problem of maximizing network lifetime for WirelessHART networks under graph routing was studied[5].

First, some researches addresses the network lifetime maximization problem of WirelessHART networks under graph routing, see the figure 1. Some contributions were made in five-fold:

• Formulation of the network lifetime maximization problem under graph routing and proof of its NP-hardness.

• An optimal network lifetime maximization algorithm based on integer programming.

•An approximation algorithm through linear programming relaxation of the integer programming algorithm.

• An efficient greedy heuristic with lower computational complexity.

• Implementation and evaluation of the proposed algorithms on a physical WSAN testbed, as well as in simulations.

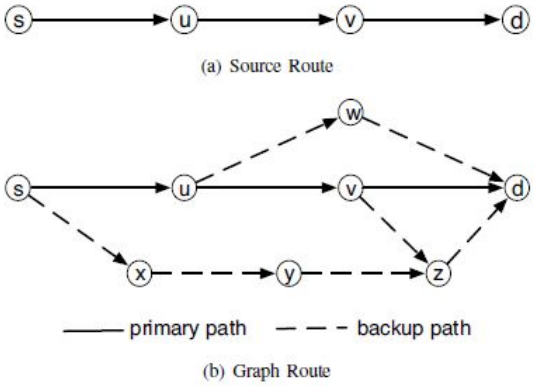

Figure 1 : Source And Graph Routing

They evaluated our routing algorithms through both experiments on a physical wireless sensor-actuator network (WSAN) testbed and simulations. It compared Integer Programming approach (IP), Linear Programming approximation (LP), and Greedy Heuristic algorithm (GH) with the reliable and real-time routing (RRC) approach that Han et al. proposed in and Dijkstra's shortest path algorithm (SP)[6].

This research evaluated routing designs on an indoor WSAN testbed consisting of 63 TelosB motes equipped with TI CC2420 radio. The testbed is located on the fifth floors of two adjacent buildings [7]. Each mote in the testbed is connected to a wired backbone network that helps facilitate the experiments and measurements without interrupting the wireless communication.

In addition, different routing algorithms through simulation in this subsection were compared. The simulator is written in C++ and follows the design of our testbed. All simulations are performed on a MacBook Pro laptop with 2.4 GHz Intel Core 2 Duo processor.

The evaluation shows that our algorithms can improve the network lifetime by up to 60%, and the greedy heuristic is more efficient than the linear programming relaxation approach.

III. RESEARCH QUESTION AND PROBLEM STATEMENT

A. The current solution:

Is using ACK/NACK messages which is a basic method that is used to check whether if the package is arrived or not. However, such a method generates an extra traffic, which causes an additional overhead that is not suitable in a highly constricted and error prone environment [8]. Therefore, another solution must be found to deal with this situation without wasting bandwidth.

B. The new solution:

Using a new reliable and energy-efficient mechanism, called AJIA (Adaptive Joint protocol based on Implicit ACK), for packet loss recovery and route quality evaluation. In this protocol, takes the advantage of the overhearing feature, which is

usually considered as a drawback, because it is consuming the battery life, and uses it for implicit ACK. By using this protocol, we ensure low latency, while minimizing the loss recovery cost by using localized data recovery among one hop neighbors [9]. Moreover, when a packet loss is detected, retransmission is carried out on the most reliable link between the node that sent the (lost) packet and its one-hop neighbors.

To achieve this protocol, we have used a different approach in comparison to traditional end-to-end error recovery mechanisms, where only the final destination node is responsible for detecting loss and requesting retransmission. Then, transform it to a hop-by-hop packet loss recovery mechanism, in which intermediate nodes also take responsibility for loss detection and recovery [9]. Intuitively, the hop-by-hop approach is more scalable and capable to recover from loss. Our objective is to provide a reliable retransmission scheme that takes into consideration the link consistency without inducing the extra overhead caused by acknowledgement messages.

Algorithm steps to implement this protocol:

1. Initialization Networking Model Phase:

Algorithm 1: Initialization Phase

Input: Plot, Nodes, and Edge.
Output: Spanning Tree.
Variables: Probability of node, node.PHistory and weight.

```
A. Sort ( Edge, Edge.weight)
B. Define Spinning Tree
C. for( i to number of Edge ) {
 add Edge(i) to Spanning Tree
 if (there is a cycle in Spinning Tree)
 dont take Edge(i)
 else
 Take Edge(i)
}
```

2. Message Relaying:

Algorithm 2: Message Relaying

Input: Spanning Tree(Nodes, Edges), Path, PacketID
Output: Packet transmitted from Source to Destination
Variables: Probability of node node.PHistory and weight.

```
A. Given Spanning Tree, Path
B. Path = list( Source, Destination)
C. i = 0
 for( node n : path ) {
 node(i).setLevel = i;
 node(i+1).setLevel =i +1;
 node(i+1).cache.add (PacketID);
 increase i
}
```

3. Lost Message Detection:

Algorithm 3: Lost Message Detection

Input: Spanning Tree (Nodes, Edges), Path and PacketID
Output: Successful or failed message passing
Variables: Packet transmission failed Failed, and waiting time TWait, time for the packet to be transmitted from on a node to other node TSpend

```
A. for( node n : path ) {
 if (node.condition == Failed) {
 status = false;
 TSpend += TWait;
 node.PHistory – ;
 print ( Packet is lost!!);
 break;
 }
 else
 status = true;
 level = node.level;
 TSpend += TWait;
 node.PHistory ++ ;
 print ( Packet successfully transmitted )
 }

}
```

4. Selective Recovery

Algorithm 4: Selective Recovery

Input: Lost Packet
Output: Select the recovery node (RecoveryNode)
Variables: Probability of node node.PHistory,
the resource of the node node.type,
the highest node reliability max, the reliability of the node Reliability,
and the node from which retransmit the packet RecoveryNode

```
A. max = 0
 for(Source → Destination){
 if(node.cache.contain(PacketID)){
 Reliability = node(i).PHistory + node(i).type;
 }
 if(Reliability > max) {
 max = Reliability;
 RecoveryNode = node(i);
 }
 }
 B.return RecoveryNode.
```

IV. IMPLEMENTATION

The simulation is done within a virtual network and sensing devices such as mobiles, laptops, and other wireless devices which are connected to this virtual network. The simulator sends the data packets from the source device to the destination device using through the virtual network. As soon as the packet is received, the simulator sends an ACK message showing the packet status. Indeed, the simulator detects the packet loss or failure by monitoring the packet transmission from one hope to another. In the situation of packet loss or failure, the simulator selects the best next hop in term of cost to process packet retransmission. The simulator continues retransmitting the packet until finally the packet arrives to the destination device.

Before progression in implementing this simulator there are some assumption must have been taken in mind. The network is composed of heterogeneous objects such as sensors, mobiles and phones. The devices within this network have different transceiver characteristics and they are randomly distributed. All the devices have sufficient resources to perform sensing, computing, and communication activities. Moreover, the data packets are randomly generated and transmitted to the sink node. The nodes are stationary during their lifetime, and they record the performance of the link between them. The transmission of the packet from the source to the sink must be completed within a specified time. If the packet does not reach the sink with this time limit, it is dropped and considered as it has been lost. Having a symmetrical channel, each two endpoints in this channel keep an identical history of the link performance. For each link, both endpoint nodes count the number of states (node up and node down).

Following the AJIA algorithm design, Java and Python are used to implement AJIA mechanism. Python NetworkX is used to establish the network and create the graph. All simulations are performed on MacBook laptop with 2.4 GHz Intel Core i7 processor. There are two setup is used for the simulations.

The first setup is done using 10 nodes with 100 * 100 as an established area which simulate a meeting room environment or a home environment . The time out for this setup is 10 seconds.
The second setup is done using 25 nodes with 200 * 200 as an established area which simulate a small working office environment. The time out for this setup is 20 seconds.
After establishing all the necessary environment, the implementation phase starts. To implement AJIA mechanism, seven steps are followed:

1.Building the network environment. This step is done by set a Plot with specific area that is defined by using the width and length boundaries. Then the nodes are Initialized with specifying the attribute of each node (id, Phis, x, y, cache, type). After that the edges between nodes are added with initialized attributes of each edge (source node, destination node, weight) as well. By the end of this step, a complete graph for all nodes and edges is created as shown in figure 2.

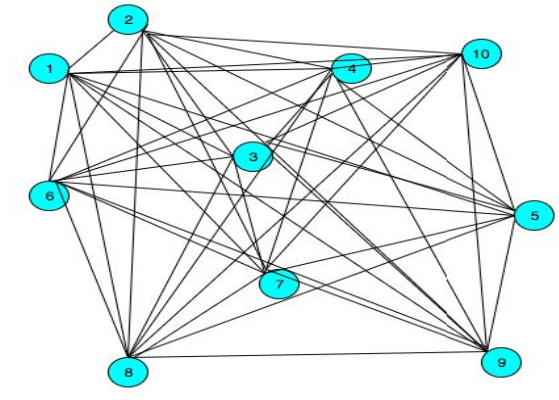

Figure 2 : Complete Graph

2.Building the Spanning Tree. This step is accomplished using the kruskal Algorithm [10]. The kruskal Algorithm is divided into steps:

a. Sorting all the edges in non-decreasing order depending on their weight.
b. Picking the smallest edge. Then, check weather this edge forms a cycle in the spanning tree or not. If cycle is not formed, then include this edge. Otherwise, discard it

Repeat this step until there are (V-1) edges in the spanning tree.

3. Getting the path for a given source node to destination node. For this step we need to use a stack and queue. The idea of using the stack is to travel through the spanning tree and push the nodes within the given path to the stack. The queue is used to discover all the neighbors nods and add them to the queue. The purpose of that is to keep tracking of the path, so we do not visit the same node again. Once the destination node is reached, the stack starts to pop the nodes out which gives us the right path from the destination node to the source. Now, the complete path from source to destination is gotten.

4. Packet relying. Simply the packet is forwarded node by node from the source node to the destination node.

5. Lost message detecting. Within a timeout limit, the source node sends the packet to the next node in the path. After sending the packet, the sending node waits to hear an ACK from the receiving node. Once the timeout and the sending node did not receive any ACK, the packet is considered lost.

6. Once the packet is lost, the simulator evaluates the node and chooses the best node in terms of costs to retransmit the packet. This evaluations is done base on two main factors:

a. The Phis which keeps a record of the the history of the links between the nods. The PHistory show the quality of the link (number of successful and failed packets transmission).
b. The resource of the node. Different resources given different characterizes. Some resources are more reliable than others. For examples, PC is more energy reliable than mobile, because PC depending on more stable energy source than mobile phone.

7. The simulator retransmits the packet from the new source (the recovery node) to the destination. The simulator keeps retransmit the packet until it reaches the final destination.

V. SIMULATION RESULT ANALYSIS

After finishing implementation of the simulator, we evaluate the AJIA mechanism by setting up two models.

In model 1, we simulated a small meeting room or home environment with 10 nodes sensing devices within 100 * 100 area. In this environment, we sent 10 different data packets from different sources and destinations by the simulator by two rounds. We also evaluate model 1 performance by sending 10 data packets from the same source node to the same destination node by 5 rounds. We compared the probability history variable of each node after sending 10 data packets with its initial value which was randomly generated initially.

In model 2, we simulated a classroom or small working office environment with 25 nodes sensing devices within 200 * 200 area. In this environment, we sent 10 different data packets from different destinations by 2 rounds. The evaluation from fixed same source and destination node was also made in model 2 by 10 rounds. Similarly, we compared the probability value of each node to evaluate whether the AJIA algorithm improve the reliability of links within this sensing networking environment.

A. Model 1 Setup Result Analysis

The model 1 environment Figure 3 (spanning tree connection for 10 sensing devices) after sending 10 different data packets from different sources and destinations by two rounds. We found that 80% of sensing devices experienced Phis increase by average 11.2% (Table 1 and 2).

In addition, we fixed the source node and destination to send 10 different data packets by 10 rounds. The results show that the average Phis increase percentage is 12.2% (Table 3), which is quite similar to the result of sending 10 data packets from different sources and destination (11.2%).

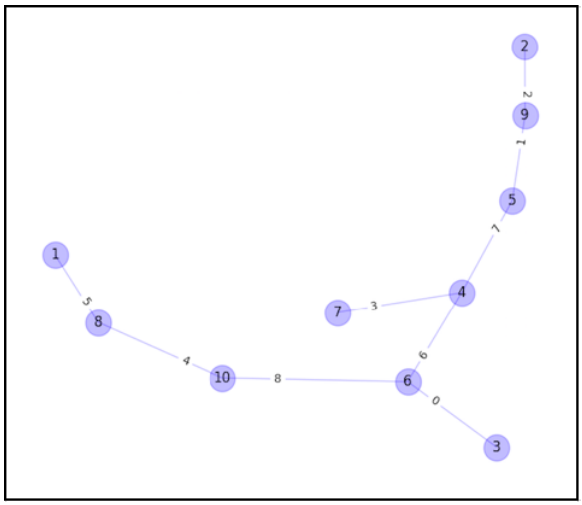

Figure 3: Model 1 Networking Environment

node id :	Phis Increase Percentage After 10 Rounds
1	0.1976
2	0.0434
3	0.0806
4	0.1598
5	0.1272
6	0.0904
7	0.1316
8	0.0728
9	0.0708
10	0.2456
AVG	0.12198

Table 3: Phis Value Change After Sending 10 Packets From Fixed Source and Destination by 10 Rounds Model 1

node id :	Phis init	Phis 1(2-6)	Phis 2(3-8)	Phis 3(4-9)	Phis 4(5-7)	Phis 5(6-1)	Phis 6(10-3)	Phis 7(9-2)	Phis 8(8-5)	Phis 9(7-3)	Phis 10(8-4)	Percentage of Increase
1	0.579	0.579	0.579	0.579	0.579	0.579	0.579	0.579	0.579	0.579	0.579	0.000
2	0.25	0.25	0.25	0.25	0.286	0.286	0.286	0.286	0.286	0.286	0.318	0.272
3	0.389	0.389	0.421	0.421	0.421	0.45	0.45	0.45	0.45	0.476	0.476	0.224
4	0.864	0.864	0.864	0.864	0.864	0.864	0.864	0.864	0.864	0.864	0.864	0.000
5	0.514	0.526	0.538	0.538	0.538	0.538	0.538	0.538	0.538	0.55	0.561	0.091
6	0.5	0.513	0.513	0.513	0.512	0.512	0.512	0.512	0.512	0.524	0.535	0.070
7	0.645	0.645	0.645	0.645	0.645	0.636	0.636	0.647	0.647	0.657	0.657	0.019
8	0.8	0.8	0.81	0.81	0.81	0.818	0.818	0.818	0.818	0.826	0.833	0.041
9	0.316	0.316	0.316	0.35	0.35	0.35	0.35	0.381	0.381	0.381	0.381	0.206
10	0.31	0.31	0.333	0.333	0.333	0.355	0.375	0.375	0.382	0.4	0.417	0.345
												0.127

Table 1: Phis Value Change After Sending 10 Packets By Different Source And Destination - 1st Round Model 1

node id :	Phis init	Phis 1(2-7)	Phis 2(3-9)	Phis 3(4-10)	Phis 4(1-8)	Phis 5(6-4)	Phis 6(9-2)	Phis 7(8-5)	Phis 8(7-1)	Phis 9(4-9)	Phis 10(10-5)	Percentage of Increase
1	0.476	0.476	0.5	0.5	0.5	0.5	0.5	0.5	0.522	0.522	0.542	0.139
2	0.5	0.5	0.5	0.524	0.524	0.545	0.545	0.545	0.545	0.545	0.545	0.090
3	0.36	0.36	0.36	0.36	0.36	0.36	0.36	0.36	0.36	0.36	0.36	0.000
4	0.462	0.462	0.462	0.462	0.462	0.462	0.462	0.462	0.462	0.462	0.462	0.000
5	0.545	0.545	0.545	0.545	0.545	0.559	0.571	0.571	0.571	0.571	0.571	0.048
6	0.429	0.429	0.429	0.429	0.444	0.444	0.444	0.459	0.459	0.474	0.474	0.105
7	0.565	0.565	0.56	0.577	0.593	0.593	0.593	0.607	0.621	0.633	0.633	0.120
8	0.591	0.591	0.609	0.609	0.609	0.609	0.609	0.609	0.609	0.625	0.625	0.058
9	0.571	0.586	0.6	0.613	0.606	0.606	0.606	0.606	0.618	0.618	0.629	0.102
10	0.607	0.607	0.633	0.645	0.656	0.656	0.656	0.656	0.667	0.667	0.676	0.114
												0.077

Table 2: Phis Value Change After Sending 10 Packets By Different Source And Destination - 2nd Round Model 1

B. Model 2 Setup Result Analysis

The model 2 environment Figure 4 (spanning tree connection for 25 sensing devices, after sending 10 different data packets from different sources and destinations by two rounds.

In table 3 and 4, it shows that 82% of sensing devices experienced Phis increase by average 5.7%.

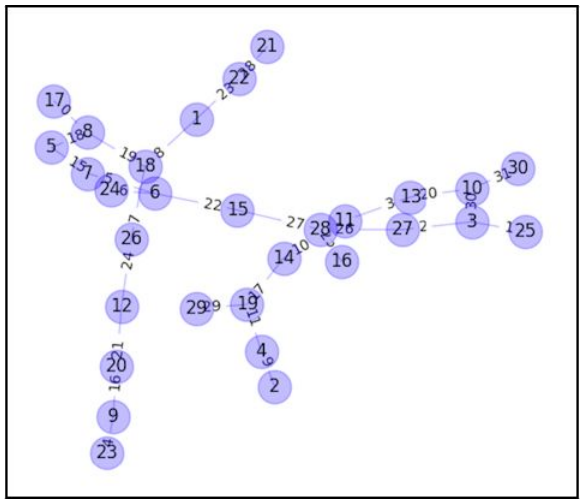

Figure 4: Model 2 Networking Environment

node id :	Phis init	Phis 1(3-23)	Phis 2(5-22)	Phis 3(9-19)	Phis 4(22-12)	Phis 5(4-25)	Phis 6(23-19)	Phis 7(20-12)	Phis 8(17-13)	Phis 9(20-9)	Phis 10(10-18)	Percentage of Increase
1	0.714	0.714	0.714	0.714	0.714	0.714	0.714	0.714	0.750	0.75	0.75	0.036
2	0.062	0.062	0.062	0.062	0.062	0.062	0.062	0.062	0.167	0.167	0.167	0.105
3	0.947	0.947	0.947	0.95	0.952	0.952	0.955	0.957	0.957	0.957	0.958	0.011
4	0.261	0.28	0.28	0.28	0.28	0.28	0.379	0.419	0.419	0.455	0.455	0.194
5	0.857	0.875	0.875	0.875	0.875	0.875	0.9	0.909	0.875	0.885	0.885	0.028
6	0.633	0.633	0.633	0.633	0.633	0.633	0.633	0.633	0.633	0.633	0.633	0.000
7	0.548	0.548	0.548	0.548	0.548	0.548	0.548	0.548	0.562	0.562	0.562	0.014
8	0.188	0.188	0.235	0.316	0.364	0.364	0.364	0.364	0.364	0.375	0.375	0.187
9	0.429	0.429	0.429	0.429	0.429	0.429	0.429	0.429	0.429	0.448	0.448	0.019
10	0.762	0.762	0.762	0.762	0.762	0.762	0.762	0.762	0.762	0.762	0.762	0.000
11	0.64	0.64	0.64	0.64	0.64	0.64	0.64	0.64	0.61	0.63	0.63	-0.010
12	0.778	0.778	0.778	0.8	0.818	0.818	0.833	0.846	0.846	0.846	0.857	0.079
13	0.696	0.696	0.696	0.696	0.696	0.696	0.696	0.696	0.708	0.708	0.708	0.012
14	0.727	0.727	0.727	0.75	0.75	0.75	0.769	0.769	0.769	0.769	0.731	0.006
15	0.476	0.476	0.476	0.5	0.5	0.5	0.5	0.5	0.5	0.522	0.522	0.046
16	0.333	0.455	0.455	0.478	0.48	0.48	0.464	0.467	0.467	0.467	0.515	0.182
17	0.462	0.529	0.556	0.55	0.591	0.591	0.64	0.667	0.667	0.69	0.719	0.257
18	0.737	0.737	0.75	0.773	0.792	0.792	0.792	0.792	0.792	0.808	0.815	0.078
19	0.28	0.28	0.28	0.308	0.308	0.308	0.333	0.333	0.333	0.333	0.333	0.053
20	0.667	0.75	0.75	0.75	0.75	0.8	0.8	0.8	0.8	0.8	0.8	0.133
21	0.375	0.4	0.417	0.417	0.417	0.417	0.45	0.476	0.5	0.522	0.522	0.147
22	0.714	0.714	0.688	0.688	0.688	0.688	0.688	0.688	0.688	0.688	0.688	-0.026
23	0.438	0.455	0.455	0.455	0.455	0.455	0.455	0.455	0.455	0.455	0.455	0.017
24	0.471	0.471	0.5	0.55	0.591	0.591	0.591	0.591	0.591	0.625	0.593	0.122
25	0.52	0.52	0.52	0.52	0.52	0.538	0.538	0.538	0.538	0.538	0.538	0.018
												0.068

Table 3: Phis Value Change After Sending 10 Packets By Different Source And Destination - 1st Round Model 2

node id :	Phis init	Phis 1(2-22)	Phis 2(3-21)	Phis 3(8-19)	Phis 4(21-12)	Phis 5(4-25)	Phis 6(20-19)	Phis 7(24-12)	Phis 8(18-13)	Phis 9(21-9)	Phis 10(10-19)	Percentage of Increase
1	0.63	0.63	0.63	0.63	0.63	0.63	0.63	0.63	0.63	0.63	0.63	0.000
2	0.528	0.528	0.541	0.553	0.553	0.564	0.585	0.585	0.585	0.585	0.581	0.053
3	0.409	0.409	0.409	0.409	0.409	0.409	0.409	0.409	0.409	0.409	0.409	0.000
4	0.462	0.462	0.462	0.462	0.462	0.462	0.462	0.462	0.462	0.462	0.462	0.000
5	0.625	0.625	0.625	0.625	0.625	0.625	0.625	0.625	0.625	0.625	0.625	0.000
6	0.531	0.531	0.531	0.531	0.531	0.531	0.531	0.531	0.531	0.531	0.531	0.000
7	0.375	0.371	0.371	0.371	0.405	0.421	0.436	0.45	0.45	0.45	0.45	0.075
8	0.606	0.618	0.618	0.618	0.611	0.622	0.632	0.641	0.641	0.641	0.641	0.035
9	0.909	0.913	0.913	0.917	0.923	0.926	0.931	0.933	0.933	0.935	0.935	0.026
10	0.5	0.529	0.529	0.556	0.6	0.619	0.652	0.667	0.667	0.68	0.68	0.180
11	0.517	0.517	0.517	0.517	0.517	0.533	0.533	0.533	0.533	0.533	0.533	0.016
12	0.905	0.905	0.905	0.905	0.909	0.909	0.909	0.913	0.913	0.913	0.913	0.008
13	0.474	0.474	0.475	0.475	0.475	0.475	0.475	0.475	0.488	0.488	0.488	0.014
14	0.818	0.818	0.833	0.833	0.833	0.833	0.833	0.833	0.846	0.846	0.846	0.028
15	0.077	0.077	0.077	0.077	0.077	0.077	0.077	0.077	0.077	0.077	0.077	0.000
16	0.091	0.13	0.13	0.13	0.2	0.231	0.259	0.286	0.286	0.286	0.286	0.195
17	0.552	0.552	0.567	0.581	0.581	0.581	0.606	0.606	0.618	0.618	0.629	0.077
18	0.5	0.5	0.517	0.533	0.533	0.533	0.562	0.562	0.562	0.562	0.576	0.076
19	0.448	0.448	0.448	0.467	0.467	0.467	0.469	0.469	0.469	0.469	0.485	0.037
20	0.5	0.5	0.5	0.5	0.5	0.512	0.512	0.523	0.523	0.523	0.523	0.023
21	0.261	0.292	0.32	0.346	0.346	0.37	0.414	0.414	0.414	0.414	0.419	0.158
22	0.464	0.483	0.483	0.483	0.516	0.531	0.545	0.559	0.559	0.559	0.559	0.095
23	0.615	0.615	0.615	0.615	0.615	0.643	0.643	0.643	0.643	0.643	0.643	0.028
24	0.053	0.053	0.053	0.053	0.053	0.053	0.053	0.053	0.053	0.053	0.053	0.000
25	0.613	0.613	0.613	0.613	0.613	0.625	0.625	0.625	0.625	0.625	0.625	0.012
												0.045

Table 4: Phis Value Change After Sending 10 Packets By Different Source And Destination - 2nd Round Model 2

Besides, we fixed the source node and destination to send 10 different data packets by 10 rounds. The results show that the average Phis increase percentage is 3.6% (Table 5), which looks like the same increase level compared to the result of sending 10 data packets from different sources and destination (5.7%).

node id :	Phis Increase Percentage After 10 Rounds
1	0.058
2	-0.001
3	0.000
4	0.034
5	0.027
6	0.021
7	0.020
8	0.030
9	0.044
10	0.000
11	0.027
12	0.032
13	0.092
14	0.044
15	0.025
16	0.045
17	0.000
18	0.105
19	0.071
20	0.018
21	0.045
22	0.016
23	0.003
24	0.092
25	0.050
AVG	0.036

Table 5: Phis Value Change After Sending 10 Packets From Fixed Source and Destination by 10 Rounds Model 2

VI. CONCLUSION

To improve the reliability of data transmission in the IoT, we implement a reliable protocol, that is AJIA mechanism, for data transmission adapted to the emergency application. Our solution is intended to operate in the IoT environment characterized by a huge diversity of devices resources by two kinds of model: 1). Meeting room or home environment within less 10 sensing devices; 2). Small working office or classroom within 25 sensing devices. The results show that AJIA mechanism is suitable to model 1 networking environment with reliability increase by average 11.2%.

For Future work, we intend to consider energy efficiency and The Link Quality Indicator (LQI) when evaluating AJIA mechanism performance in the aspect of reliability.

VII. REFERENCE

[1] L. Yang, S.H. Yang, L. Plotnick, "How the internet of things technology enhances emergency response operations", Technological Forecasting & Social Change, 2012.

[2] D. Giusto, A. Iera, G. Morabito, L. Atzori (Eds.), The Internet of Things, Springer, 2010. ISBN: 978-1-4419-1673-0.

[3] M. A. Razzaque, M. Milojevic-Jevric, A. Palade, and S. Clarke, "Middleware for internet of things: a survey,"IEEE Internet of Things Journal,vol.3,no.1, pp.70–95,2016.

[4] M. Nassr, J. Jun, S. Eidenbenz, A. Hansson "Scalable and reliable sensor network routing: Performance study from field deployment," in The 26th IEEE INFOCOM, pp. 670–678, 2007.

[5] J. Song, S. Han, A. K. Mok, D. Chen, M. Lucas, and M. Nixon, "WirelessHART: Applying Wireless Technology in Real-Time Industrial Process Control," in IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS'08), April 2008.

[6] C. Lu, A. Saifullah, B. Li, M. Sha, H. Gonzalez, D. Gunatilaka, C. Wu, L. Nie, and Y. Chen, "Real-Time Wireless Sensor-Actuator Networks for Industrial Cyber-Physical Systems," Proceedings of the IEEE, 2016.

[7] G. Gao, H. Zhang, and L. Li, "A Reliable Multipath Routing Strategy for WirelessHART Mesh Networks Using Subgraph Routing," Journal of Computational Information Systems, vol. 9, 2013.

[8] Y. Xiao, X. Li, Y. Li, and S. Chen, "Evaluate reliability ofwireless sensor networks with OBDD," in IEEE International Conference on Communications (ICC '09), pp. 1–5, 2009.

[9] Maalel N, Natalizio E, Bouabdallah A, Roux P, Kellil M. Reliability for emergency applications in internet of things. InDistributed Computing In Sensor Systems (DCOSS),International Conference On 2013 May 20 (pp. 361-366). IEEE.

[10] T. H. Cormen, C. Stein, R. L. Rivest, and C. E. Leiserson, Introduction to Algorithms, 2nd ed. McGraw-Hill Higher Education, 2001.