DIWSAN: Distributed Intelligent Wireless Sensor and Actuator Network for Heterogeneous Environment

Cheng-Min LIN††, Member, Jyh-Horng LIN†††, and Jen-Cheng CHIU†††, Nonmembers

SUMMARY In a WSAN (Wireless Sensor and Actuator Network), most resources, including sensors and actuators, are designed for certain applications in a dedicated environment. Many researchers have proposed to use gateways to infer and annotate heterogeneous data; however, such centralized methods produce a bottlenecks networking and computational overhead on the gateways that cause longer response time in activity processing, worsening performance. This work proposes two distribution inference mechanisms: realization and sequential inference mechanisms to reduce the response time in activity processing. Finally, experimental results for the proposed inference mechanisms are presented, and it shows that our mechanisms outperform the traditional centralized inference mechanism.

key words: wireless sensor and actuator network, distributed system, heterogeneous environment, inference mechanisms

1. Introduction

As next-generation computing platforms, WSNs (Wireless Sensor Network) are being applied to an increasing range of areas, such as health care improvement, environmental surveillance, climate research, ecological forestry and wildlife monitoring. WSN is a self-organizing network and consists of about one thousand of sensor nodes for collecting data. Initially, such WSNs were used with a single type of sensors deployed in a large geographic area. For example, it can be used to detect forest fire. With the development of various types of sensors, WSNs are now used in many civilian applications. For example, it was used in habitat monitoring, healthcare, home automation and traffic control [10], [19]. A WSAN (Wireless Sensor and Actor Network) or Wireless Sensor and Actuator Network [20]) is composed of sensing nodes and actuators. It is an extension of WSNs. Both information flows exist in WSANs. One is many-to-one information flow to represent sensors for providing data to the gateway. The other is one-to-many information flow to indicate that actuators execute specific actions in response to the data by processing and inference. Sensors and actuators must be efficiently coordinated and the deadline for response requirements is limited. To achieve this objective, coordination mechanisms must be implemented carefully for sensor and actuator nodes by assigning resources optimally to reduce the response time to meet the deadline. In addition, the heterogeneous sensor network is important. Many sensing devices such as thermometers, hygrometers and location sensors, for example, and many actuating devices, such as TVs, window controllers and air conditioners are in smart homes, smart offices and other environments. These are heterogeneous devices in WSN/WSAN which are a part of everyday life. ZigBee is a low-cost, low-power, wireless network standard [23]. The low cost allows the technology to be widely deployed in wireless control and monitoring applications. The low power-usage allows longer life with smaller batteries. ZigBee is very easy to add to various devices. Hence, it is treated as very suitable for WSAN communication technologies in heterogeneous environments.

In [2], [21], data from sensor is gathered on a sink or a gateway for processing and inference. Such mechanism is called centralized inference mechanism. In many researchers, centralized inference algorithms have been used in the sensor networks, such as belief propagation [6], [16] and particle filtering [5]. Several problems associated with the centralized approach must be solved. First, the approach produces a bottleneck node at the gateway. Since sensory data from all sensors are transferred to the gateway, the network is congested with sensory data packets. When the network size becomes large and transmission rate is low such as the ZigBee network, the problem becomes more serious. Secondly, the gateway must be powerful to compute the data from all sensors that the cost of the gateway will be high. When the network size becomes large, the processing time will increase since more data are required to process and the transmission will become long since the distance between far sensor and gateway becomes large. In this work, DIWSAN uses distributed inference approach to solve above problems. However, extensible Markup Language (XML) is used to describe heterogeneous devices, data [7], [8] and activity conditions for integrating heterogeneous devices. ZigBee involves two types of devices - FFD (Full Function Device) and RFD (Reduced Function Device). FFD can communicate with both FFD and RFD and it can be a coordinator, router or end device. RFD can only communicate with FFD and it only can be an end device. Therefore, RFD re-
quires fewer resources, including memory size. The FFD has a larger memory capacity for implementing inference mechanism. The distributed inference is implemented in ZigBee-based node, which is technology commonly using in sensors and actuators to reduce the response time for solving the centralized inference problem. Our proposed method can be applied in WSAN domains such as healthcare, home automation, and agricultural monitoring etc.

The rest of this paper is organized as follows. Section 2 introduces related works and the background of ZigBee. In Sect. 3, the overview and system architecture of DIWSAN are presented. Section 4 presents system information used in DIWSAN and Sect. 5 discusses the inference mechanisms. Section 6 describes the system implementation and prototype. Simulation results are showed in Sect. 7. Finally, Sect. 8 addresses some conclusions.

2. Related Work

Related works cover the component techniques of heterogeneous data description, distributed inference and ZigBee. Several works[12],[13] have proposed for specific applications of WSN, such as parking-lot networks. They are all data-centric schemes, in which all communications are for sensor involves named data. The work[18] proposed sensor networks that includes heterogeneous sensor devices and supports a large range of applications. The article[7] proposed a Web services approach for the design of sensor network, in which sensor nodes are service providers and applications are clients of such services. Distributed inference [17] is one of the primary applications of WSAN. Such inference includes aggregation, detection, estimation, and control action. In the work[14], the distributed decision fusion rules are used in sensor node to reduce communication time. In the article[15], a distributed architecture is proposed to solve probabilistic inference, regression, and control problems in an unreliable communication environment. The type-based random access scheme [3] discusses the problem of distributed Bayesian estimation. Two schemes, centralized and distributed controls, have been proposed in [4] to control actuators. Their simulation results reveal that distributed control assures similar steady as central control while distributed control less sensitive towards packet loss than centralized control. In these research efforts, lacks of the cited works have addressed simultaneously heterogeneous data and the distributed inference problem.

3. System Overview

This section introduces DIWSAN that adopts the original WSAN hardware architecture and extends the architecture by adding smart nodes in heterogeneous environment for improving traditional centralized inference performance. In Sect. 3.1, we introduce detail system architecture and describe system data flow chart in Sect. 3.2.

3.1 DIWSAN Architecture

This section introduces the system. Figure 1 shows the system architecture of DIWSAN, including relevant information and network configuration. DIWSAN adds several special nodes, called smart nodes. In addition to aggregating sensory data from neighbor nodes, each node can process data and inference results based on the relevant information. Each smart node exploits this information to recognize data from neighboring nodes and conform to the conditions of activities. The information consists of activities and information of the node. Node information includes sensor or actuator resource, vendor, function and other properties. Activity information is expected to lead to corresponding behavior by the user. An activity includes conditions and actions. A condition may be an expression of sensing data or actuator status. For examples, the temperature exceeds 20 degrees or a window is open. An action is a particular behavior such as closing the window when the user-defined conditions are satisfied. A user can build up any combination of system nodes to form an activity to meet his/her individual requirements.

DIWSAN provides two novel inference mechanisms. DIWSAN architecture is defined as follows.

Definition 1. DIWSAN is defined as a 3-tuple DIWSAN = [M, N, SI], to which the following apply.

- M denotes one of three inference mechanism models - centralized inference mechanism (CIM), regionalized inference mechanism (RIM) and sequential inference mechanism (SIM). Detailed specifications of them will present in Sect. 5.
- N denotes a set of nodes with four types - gateway, sensor, actuator and smart node.
- SI is used to present system information in an inference mechanism. The SI includes two parts, node information (ni) and activity information (ai). It will be discussed in Sect. 4.

Definition 2. The hardware components of DIWSAN are defined as a 4-tuple N = (G, S, A, I).

- G denotes a gateway which is employed to integrate the
outside and inside network. The node stores clear and meaningful system information such as user-defined system activity and node information. An XML compiler can encapsulate ai in an XML data structure. An XML parser can decode node and sensing data from a WSAN. XML is used to describe heterogeneous data from sensors and actuators. The goal is for the smart node to recognize data at the neighboring nodes.

- \( S = \{s_1, s_2, \ldots, s_n\} \) is a set of sensors. A sensor is always an input device. The purpose of a sensor is to detect the status of environmental condition. It can be deployed in a heterogenous environment to detect changes, such as temperature, humidity, light, smoke, and so on. Once a sensor detects any changes, it soon delivers the sensing information to an appropriate node to determine system activities.
- \( A = \{a_1, a_2, \ldots, a_m\} \) is a set of actuators. An actuator is an output type device such as an air conditioner, television, radio, refrigerator or ice box, for example. An actuator node plays a passive role in a heterogenous environment. Once an activity condition is satisfied, the activity defined in the action field of an actuator node will be triggered for suitable behavior.
- \( I = \{i_1, i_2, \ldots, i_p\} \) is a set of smart nodes which are special nodes that have inference ability. Actually, a smart node is either a sensor or an actuator. The smart node stores the user-defined system activities. The responsibility of the smart node is to collect sensor events, process events and trigger some actions defined in activities.

3.2 DIWSAN Data Flow Chart

Figure 2 shows a flow chart for system message exchange. When sensors and actuators are deployed in a heterogeneous environment, these nodes transmit their ni to the gateway using XML. After the gateway has recognized all nodes, the system can create activity defined by user. Then, the smart nodes will be picked out from the sensors and actuators of related activity. The gateway sends ni to the smart node and tells it to play an inference role. The sensors whose activity conditions need sensory data will be notified by the smart node. The actuator presented in an activity action will be notified which smart node can control it. The gateway will send ai to notify the smart node of its task, including the aggregation of sensory data and inference, to trigger the actuators. After the smart node receives ai, it can recognize sensory data from neighboring nodes. If the activity conditions are satisfied, some actions are triggered and the smart node notifies the actuator to act.

4. System Information

In the heterogenous environment, the gateway must record the ID, type and location of each node and transfers SI and raw data. However, the smart node cannot store the raw data format of every node because the node memory is not enough. Therefore, each node uses XML to describe individual characteristics in the system. The nodes only store XML format and can recognize data from neighboring nodes. This section defines the XML tags and attributes.

The definition of all system information is presented by using XML, which is widely used to represent and exchange structured data over the Web. SI includes: \( ni \) and \( ai \). An individual \( ni \) exists in every node, and includes \( S \) and \( A \), it is used to enable the inference mechanism. The attributes of \( ni \) are described by XML and are defined below.

\[ ni = \{nodeprofile, node, data, location, address\} \]

Each node must follow the format in the deployed environment.

- \(<nodeProfile>\) represents the \( ni \) profile.
- \(<node>\) defines the basic information about the node. The tag has three attributes. The “type” attribute indicates whether the node is a sensor or an actuator. The “vendor” attribute is the name of the device vendor. The “description” attribute gives a brief explanation of the node. The node can be recognized by this information at a gateway.
- \(<data>\) represents input/output data of the node device. The tag consists of two attributes. The “type” attribute defines the data format of the node; the “description” attribute is data information.
- \(<locationId>\) represents node location identifier. The tag contains one attribute. The “name” attribute defines the location of the node device.
- \(<address>\) represents the physical network address of node device. The tag is especially important to the proposed architecture. Using this tag, the node informs the correct address of environmental changes.

An ai defines user activity. Each node should know the predefined XML tag and able to parse the user activities in the system. The ai attributes are defined as \( ai = \{c, ac\} \), in while \( c \) is a set of conditions and \( ac \) is a set of actions.

- \(<activityProfile>\) defines user activity.
- \(<condition>\) defines the system status that is expected by the user.
- \(<action>\) is the behavior of system if the conditions are satisfied.
Figure 3 shows an example of activity and node information. It shows temperature regulating activity. The activity condition is \(( \text{sensor0001} > 28.5 \land \text{sensor0002} > 60) \lor (\text{sensor0003} = 1) \Rightarrow (\text{actuator0012} = 1 \land \text{actuator0013} = 1)\). Actuator0012 and actuator0013 are a window controller and an air conditioner, respectively. If the conditions associated with either sensor001 and sensor002 or sensor0003 are satisfied, then the two actuators can be opened to open a window and turn on the air conditioner.

5. Inference Mechanisms

This section introduces three inference mechanisms, CIM, RIM and SIM. In a smart home, the environment deployed various devices is similar to a heterogeneous sensor network. In addition to the gateway, the heterogeneous environment consists of three types of devices - \(S\), \(A\) and \(I\). The detail specification is introduced in Definition 2. ZigBee is a commonly used communication scheme in the heterogeneous environments. ZigBee is employed herein to design and implement the proposed inference mechanism and establish a ZigBee-based WSAN.

Two types of physical ZigBee devices exist - FFD and RFD. The main differences between them are in computing ability and memory. The FFD has more functions and more memory, since it is always a router in a ZigBee network. According to the requirements of computing ability and memory, the activities of the system will be proceed on an FFD as a smart node for logical inference in the proposed architecture. Two inference mechanisms, regionalized and sequential will apply in the environment and Fig. 4 shows data flow chat of the mechanisms. The regionalized mechanism consists of a few FFDs and more RFDs. The smart node is always waiting for sensor data sent by sensor nodes to meet the condition. An action profile consists of two parts, conditions and actions. All sensory data are collected to inference in a smart node. All action in the action profile will be triggered while all conditions in the profile are met as shown in Fig. 4(a). In the proposed sequential mechanism, all nodes are FFDs. Each node can be treated as a smart node. It responds immediately when any condition is fired in any node. Then, the sensory data are then transmitted to next node to inference until all of the conditions are met and the actuators are triggered as shown in Fig. 4(b). In these two schemes, many specified actuators in an action profile, they will be triggered if all conditions in the profile are satisfied. From the perspective of system execution, the RIM is a passive methodology. In contrast, the SIM is active.

5.1 Centralized Inference Mechanism

CIM is a traditional inference mechanism. In this heterogeneous environment, each sensor is a transferred device and each actuator is a received device. All sensing data are delivered to the gateway which manages all sensors and actuators and stores all nodes and ai. Some activities are inferred by the gateway. Figure 5 shows an example of a centralized inference mechanism which involves sensors (as
s1, s2, s3, s4, s5, s6, ... and actuators (as a1, a2 ...). s1 and s2 deliver sensing data to the gateway via s5. Sensor s3 delivers sensing data to the gateway via s6 and s5. In this method, s4 delivers sensing data via s7, s6 and s5. In contrast, when the gateway detects some changes, the control data are delivered to a1 and a2 via s5, s6 and s7. s5, s6 and s7 must be a FFD; the others are not limited.

5.2 Regionalized Inference Mechanism

RIM has several smart node resources in a heterogonous environment. The condition defined in the system activity will be transferred to the inference rule, in a process that can be treated as regional management. In regional management, a virtual community consists of several sensors and actuators that are managed by a smart node. Thus, the number of nodes in a region is up by a user. Example of the RIM is shown in Fig. 6.

The system includes gateway, smart nodes, sensors and actuators. The user sets three system activities into i1, i2 and i3, respectively which manage the virtual community that is surrounded by the dotted line. As shown in the Fig. 6, the devices that are managed by i1 are not only one hop away (s1, s2) but also multi-hop away (s3). Different virtual communities can include the same nodes. In regionalized mechanism, most of the nodes are RFDs. Hence, a sensor in the RFDs can only read simple sensory data from environment and an actuator in the RFDs can only output some control signals to peripherals. Only a few nodes can act as smart nodes. When the sensor node detects some changes, it simply transmits the changes to the corresponding smart node; for example, s2 transmits to i1; s4 transmits to i3; s3 transmits to i2 and i1. Once all conditions of an activity in a smart node are satisfied, the smart node instructs the actuator device to output the control signal to its peripheral, such as i1 to a1, and i3 to a2.

5.3 Sequential Inference Mechanism

SIM is an active scheme that depends on FFD hardware for each node in a heterogonous environment. In the system, each node is a smart node because using the mechanism requires that every node has high computation ability and a large memory for storing system activities. Hence, every node plays an inferential role as a smart node. Figure 7 shows an example of the SIM. The system involves three activities, which are surrounded by the dotted line in the system. An activity is represented by the left dot line; the same conditions of the activity are copied into each node (i1, i2, i3, i4 and i5). The condition status will be transmitted to the next node, whenever a node detects environmental changes. For example, the condition rule first checks whether node i3 detects an environmental change via its sensory data. If the condition changes satisfy the rule that is defined in node i3, then the node transmits the condition to node i1. The node i1 checks its sensory data immediately. It stops transmitting the condition to the next node if the condition is not satisfied. Otherwise, it transmits the conditions to the next node. The next node follows this decision rule to determine whether the condition should be transmitted to the following node.

6. Implementation

This section describes the implementation of DIWSAN. First, Sect. 6.1 introduces the block and functionality of the implemented hardware node. Section 6.2 describes the software layers on the node. Section 6.3 describes the GUI for defining the system activities.

6.1 Hardware of DIWSAN Node

Figure 8 shows the implemented ZigBee platform. The microcontroller is an Atmega64L, produced by Atmel Incorporated. The Atmega64L is a high-performance 8 bit RISC microcontroller with 64 K bytes of in-system reprogrammable Flash and 2 K bytes EEPROM. The operating voltage is between 2.7 and 5.5 V. Because of low power and low resource requirements of ZigBee, the peripheral of Atmega64L contains two 8 bit timers, an 8-channel 10-bit ADC, dual programmable serial USARTs and a Master/Slave SPI serial interface. These peripheral features reduce extra peripheral circuits to be designed. Thus, the plat-
form can be narrowed shrunk as much as possible. An XBee OEM RF module [22] produced by MaxStream Inc. is selected as the IEEE 802.15.4 RF chip. XBee supports two operation modes - transparent and application programming interface (API). When the XBee module operates in transparent mode, a set of AT commands is supported to control the internal ZigBee stack. Using the AT command, a user can quickly establish an application without the detailed knowledge of ZigBee stack. The API mode defines a serial communication protocol to act as a pure IEEE 802.15.4 RF module. Application should be implemented in one of two operation modes of ZigBee stack.

6.2 Software of Smart Node

Figure 9 shows the software function block of implemented smart node. The software associated with the node was realized using the WinAVR [25]. WinAVR is a suite of executable, open source software development tools for the Atmel AVR series of RISC microprocessors hosted on the Microsoft Windows platform. FreeRTOS [24] is a portable, open source, mini Real Time Kernel. It is designed to be small, simple and easy to use. The ZigBee protocol stack is embedded in XBee. AT commands exploit communication between XBee and MCU via a serial port. The user-defined system activities are encoded in XML. The XML parsing block is responsible for transforming activities into a logical expression to enable the rule judge block to determine how to respond to the received message. The GPIO and ADC drivers directly control the peripherals in MCU to provide an interface for communication with the external sensor or actuator. The data sensory and action trigger blocks depend on the hardware chip.

6.3 Software of Gateway

Users should be able to construct system activities simply using a universal method. The personal computer is adopted as a gateway with a graphic user interface. The gateway GUI has two purposes. The first is to parse the XML message into meaningful information. The second is to construct user-defined system activities in XML and update the node that is selected by the users. Figure 10 depicts the program for implementing the GUI. The GUI was designed with five blocks - System Node, Management, Activity Information, Node Information and Condition Information.

**System Node** block exhibits all displayed nodes in DIWSAN. When any new node joins the network, its name is automatically added to the list. The user can select any node in the list and associate activities with the selected node. **Management** block includes three operating buttons - add activity, delete activity and update to node. When a user selects a node in the system node block, the activity can be added to or deleted from the selected node using the appropriate button. Once all activities have been added to the node, the user can update these activities to the selected node using the ‘update to node’ button. **Activity Information** block is used to display the activities of the node. User can add a definition to the condition or action tag. **Node Information** block is used to display detailed information of a node. When a user selects a node on the ‘device select’, the other filed will show the information about the selected node, such as vendor, node name and others. **Condition Information** block is used for adding or removing a description of behavior in the selected rule. User can express the data or action of a selected sensor/actuator in the Node Information block. The ‘add behavior description’ button is used to add a behavior of the selected rule. The button ‘remove selected behavior’ is used to delete a behav-
ior from the selected rule.

7. Experimental Results

This section presents simulate results concerning the performances of the three inference mechanisms. The main goal is to test response time of the sequential, regionalized mechanism as well as CIM in WSANs in the same environments. The response time is defined to calculate the total run time for aggregating data, rules decision and actuating devices. Table 1 presents the related simulation parameters. These simulated environments are constructed using ZigBee star topology. The Zigbee of Contention Free Period (CFP) protocol is used to our simulator. It is no congest protocol. In the simulated environments, all three inference mechanisms consist of 20 or 50 smart nodes that are randomly deployed into a network. A smart node is deployed to the center of a virtual community. The neighbor nodes are assumed one hop away with smart node in simulation network. The transmission range is assumed 10 m using the implemented devices to actually measure between FFD and RFD. The connection between all sensors and actuators is randomly assigned. One simulation scenario is executed 500 times. The response time is measured as an average of time of completion all activities.

Some assumptions are made in the simulation. First, all nodes in the network are static. Second, all RFDs connected to a single FFD form a system activity. Third, events were generated periodically within a designed time in each node. Fourth, an event packet is generated from a sensor and the length of the packet is fixed at 30 bytes. For data transmission rates of 250 Kbps, 40 Kbps and 20 Kbps, the event transmission time is nearly 1 ms, 6 ms and 12 ms respectively.

7.1 Response Time for Various Data Rate

In this section, several event sensing times are simulated at various data rate. The sensing period is varied from 300 ms to 850 ms. The network parameters, including the number of FFDs, the number of RFDs and the number of system activities are fixed at 50, 500 and 30, respectively, in the simulated networks. The 30 system activities are randomly assigned to 50 smart nodes.

Figure 11 shows the response time of three inference mechanisms at a transmission rate of 250 Kbps. SIM has the best response time because activity processing in this mechanism is active methodology. While any event is generated by a node, the node immediately determines based on its stored conditions, whether to forward the activity status to the next node. CIM and RIM both have to collect data first to conduct inference. Therefore, the event sensing period and average response time are increase proportionally. The response time of RIM is less than 15% of that of CIM because RIM processes events is parallel. In RIM, the event is transmitted only to the corresponding FFD for decision action. However, RIM gets a better response time than CIM.

Figure 12 shows the simulation result for a data transmission rate of 40 Kbps. The results are similar to those at 250 Kbps. Since the transmission rate is lower and sensor period is shorter, collecting data require a longer time in CIM. Therefore, CIM has a longer average response time. However, RIM outperforms CIM by 20%.

Figure 13 shows the simulation results at a data transmission rate of 20 Kbps. When the sensing period is less than 600 ms, the response time is almost same in CIM, because the network bandwidth cannot handle the generated

| Table 1 | Simulation parameter. |
|---------|-----------------------|
| Parameter | Values               |
| Environment Area | 1000m²               |
| Transmission Range | 10 m                |
| The Number of Smart Nodes | 50 nodes          |
| The Number of Activity Rule | 30 rules          |
| Node Placement | random deployment    |
| Packet Size     | 30 bytes             |
| Data Transmission Rate | 20, 40, 250 Kbps |
| Simulation Rounds | 500 rounds         |

![Fig. 11](image1.png) Response time at transmission rate of 250 Kbps.

![Fig. 12](image2.png) Response time at transmission rate of 40 Kbps.
network events. Any event should be finally transmitted to the gateway for a decision. However, all events are received on the ZigBee coordinator area, indicating that the bandwidth near the coordinator range is very congested. However, both regionalized and sequential inference mechanisms can handle the generated event in a reasonable time.

7.2 Response Time under Various Conditions

In this section, the effect of number of conditions was simulated. The numbers of FFDs, event sensing periods and system activities are fixed. There are 15 system activities are randomly assigned to 20 FFDs in the simulated network. The event sensing period is fixed at 200 ms. The number of conditions is varied from 100 to 500 units and randomly allocated into 15 activities.

Figure 14 shows the effect of different numbers of conditions at a data transmission rate of 250 Kbps. When the number of conditions is less in SIM, deployed RFDs in the simulator are loose and produce a greater distance between nodes. Hence, the response time is unstable. Although SIM processes events from one node to another, the bandwidth is enough to handle the generated event. Therefore, SIM has the best response time of all mechanism. The response time of CIM is exceeds 200 ms when the number of conditions is exceeds 200. Hence, the network bandwidth cannot handle the generated events. However, the generated events can still be handled in RIM.

Figures 15 and 16 plot the response times at two low transmission rates, 40 and 20 Kbps, respectively. In RIM, all RFDs connect to the smart node by one hop, so conditions increase but do not increase the response time. In this experimental, the event sensing period is fixed at 200 ms. Hence, increasing the conditions can make the response time approximate 200 ms. SIM must depend on the number of conditions to determine whether an action should be triggered. For example, an activity has ten conditions, SIM should then transmit ten times to determine whether the conditions are matched. The response time increases with the number of conditions. For this reason, the response time of SIM with 300 conditions exceeds that of RIM as shown in Fig. 16. This result can be roughly calculated. The average number of conditions in one activity is nearly 18. The transmission rate 20 kbps is used to transfer data. The packet size is fixed at 30 bytes. The transmission time is nearly 12 ms (30 × 8/20k). The time cost is near 216 ms (18 × 12 ms).
7.3 Summary

These three simulated results support a conclusion that holds in a low data transmission rate environment. The communication link cannot transfer many data and the sensor network causes a congestion problem, because many packets must be immediately sent. Therefore, many collisions occur in CIM that has no effect when the network transmits raw data at a low rate. However, general sensor networks work in a low transmission rate environment. As the number of conditions increased, the response time increased in CIM but the average response time in RIM was nearly the sensing period. However, the number of conditions influenced the response time in SIM.

Compare the cost of three schemes, RFD requires fewer resources than FFD, including CPU and memory size etc. Therefore, RFD is cheaper than FFD for deploying sensor network. In SIM, all devices are FFD to WSAN. CIM and RIM both have fewer smart nodes to be deployed in the network. If excluding the costs of building gateway, SIM scheme has more cost than the others.

8. Conclusions

This work presents two novel inference mechanisms, regionalized and sequential, in DIWSAN. Distributed inference is used to reduce the response time of action and is implemented in all smart nodes. It can distribute aggregate sensor data to reduce the data transmission time and to delete the bottleneck of the gateway. To re-use system resources, XML is used to describe node resources and enable sensors and actuators to recognize each other easily in a heterogeneous environment. Then, a user can dynamically configure system activities according to his/her expectations using a GUI. RIM is a passive methodology which processes system activities in parallel on different smart nodes. SIM is an active methodology which immediately reacts when a system event occurs. The experimental results show that both the regionalized and sequential inference mechanisms provided a response time that is shorter than CIM. In the future, the web technology will be used to parse XML and present sensor data.

Acknowledgements

The authors would like to thank the National Science Council of the Republic of China, Taiwan for financially supporting this research under Contract NSC 97-2221-E-252-016.

References

[1] I.F. Akyildiz and H.H. Kasimoglu, “Wireless sensor and actor networks: Research challenges,” Elsevier Journal on Ad Hoc Networks, vol.2, no.4, pp.351–367, Oct. 2004.
[2] I.F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, “A survey on sensor networks,” IEEE Commun. Mag., vol.40, no.8, pp.102–114, Aug. 2002.
[3] A. Anandkumar, L. Tong, and A. Swami, “Distributed estimation via random access,” IEEE Trans. Inf. Theory, vol.54, no.7, pp.3175–3181, July 2008.
[4] X. Cao, J. Chen, Y. Xiao, and Y. Sun, “Control systems designed for wireless sensor and actuator networks,” Proc. IEEE International Conference on Communications, pp.4968–4972, May 2008.
[5] M. Coates, “Distributed particle filtering for sensor networks,” Proc. 3rd International Symposium on Information Processing in Sensor Networks, pp.99–107, April 2004.
[6] C. Crick and A. Pfeffer, “Loopy belief propagation as a basis for communication in sensor networks,” Proc. Uncertainty in Artificial Intelligence, July 2003.
[7] F.C. Delicato, P.F. Pires, L. Pirmez, and L.F. Carneiro, “A service approach for architecting application independent wireless sensor networks,” Cluster Computing, vol.8, no.2-3, pp.211–221, July 2005.
[8] X. Du and F. Lin, “Maintaining differentiated coverage in heterogeneous sensor networks,” EURASIP Journal on Wireless Communications and Networking, vol.2005, no.4, pp.565–572, March 2005.
[9] J. Gehlke and S. Madden, “Query processing in sensor networks,” IEEE Pervasive Computing, vol.3, no.1, pp.46–55, Jan. 2004.
[10] S. Hadim and N. Mohamed, “Middleware challenges and approaches for wireless sensor networks,” IEEE Distributed Systems, vol.7, no.3, pp.1–23, March 2006.
[11] J. Hill, R. Szewczyk, A. Woo, S. Hollar, D. Culler, and K. Pister, “System architecture directions for networked sensors,” Proc. 9th International Conference on Architectural Support for Programming Languages and Operating Systems, pp.93–104, Nov. 2000.
[12] C. Intanagonwiwat, R. Govindan, and D. Estrin, “Directed diffusion: A scalable and robust communication paradigm for sensor networks,” Proc. ACM/IEEE International Conference on Mobile Computing and Networking, pp.56–67, Aug. 2000.
[13] J. Kulis, R.B. Heinzelman, and H. Balakrishnan, “Negotiation-based protocols for disseminating information in wireless sensor networks,” ACM Wireless Networks, vol.8, no.2/3, pp.169–185, March 2002.
[14] R. Niu and P.K. Varshney, “Distributed detection and fusion in a large wireless sensor network of random size,” EURASIP Journal on Wireless Communications and Networking, vol.5, no.4, pp.462–472, Sept. 2005.
[15] M. Paskin, C. Guestrin, and J. McFadden, “A robust architecture for distributed inference in sensor networks,” Proc. IEEE Fourth International Symposium on Information Processing in Sensor Networks, pp.55–62, April 2005.
[16] K. Pfarre and P.R. Kumar, “Extended message passing algorithm for inference in loopy Gaussian graphical models,” Elsevier Ad Hoc Networks, vol.2, no.2, pp.153–169, 2004.
[17] H.V. Poor, “Distributed inference in wireless sensor networks: Some recent results,” Proc. IEEE/SP 13th Workshop on Statistical Signal Processing, pp.1188–1189, July 2005.
[18] H. Qi, P.T. Kuruganti, and Y. Xu, “The development of localized algorithms in wireless sensor networks,” MDPI Sensors, vol.2, no.7, pp.286–293, July 2002.
[19] K. Romer and F. Mattern, “The design space of wireless sensor networks,” IEEE Wireless Communications, vol.11, no.6, pp.54–61, Dec. 2004.
[20] R. Verdone, D. Dardari, G. Mazzini, and A. Conti, “Wireless sensor and actuator networks: Technologies, analysis and design,” Elsevier Science & Technology Books, April 2008.
[21] J.E.L. Vergara, V.A. Villagrá, C. Fadón, J.M. González, J.A. Lozano, and M. Álvarez-Campana, “An autonomic approach to offer services in OSGi-based home gateways,” Comput. Commun., vol.31, no.13, pp.3049–3058, Aug. 2008.
[22] MaxStream Incorporated, Xbee RF Module Product Manual v8.x17 Beta - ZigBee Protocol, http://www.c-stamp.com/Documents/Documentation/CS47300Xm2.pdf
[23] ZigBee Specification 2006, ZigBee Alliance, http://www.zigbee.org/en/index.asp
[24] Free RTOS, The Standard Solution For Small Embedded System. http://www.freertos.org/

[25] WinAVR development tool, http://winavr.sourceforge.net/index.html

Cheng-Min Lin was born in 1964. He received the B.S. and M.S. degrees in electronic engineering from National Taiwan University of Science and Technology, Taipei, Taiwan, in 1989 and 1991, respectively, and the Ph.D. degree in Department of Information Engineering and Computer Science, Feng-Chia University, Taichung, Taiwan. Currently, he is an Associate Professor in the Department of Computer and Communication Engineering, Graduate Institute of Electrical Engineering and Computer Science, Nan Kai University of Technology. His research interests include embedded systems, mobile computing, distributed systems, and telematics. Dr. Lin is a member of the IEEE Computer Society.

Jyh-Horng Lin was born in 1975. He received the B.S., M.S., and Ph.D. degrees in information engineering from Feng Chia University, Taiwan, in 1998, 2000, and 2005, respectively. Currently, he is an Assistant Professor in the Department of Electrical Engineering, Nan Kai University of Technology, Taiwan. His research interests include mobile computing, ad-hoc wireless networks, distributed systems, and computer algorithms.

Jen-Cheng Chiu was born in 1974. He received the M.S. degree in Information Engineering and Computer Science from Feng-Chia University, Taichung, Taiwan, in 2004. Currently, he is a candidate for the Ph.D. degree in the Department of Information Engineering and Computer Science, Feng-Chia University, Taichung, Taiwan. His research interests include e-learning and embedded systems.