DE GRUYTER
OPEN

JACSM 2016, Vol. 8, No. 1, pp. 41 - 67
10.1515/jacsm-2016-0004

RESOURCE MONITORING FOR WIRELESS SENSOR NETWORKS USING ANFIS

Nagesha[1], Sunilkumar S. Manvi[2]

[1] Department of Electronics and Instrumentation Engineering
JSS Academy of Technical Education, Bengaluru, India
nageshashiva@gmail.com

[2] School of Computing and Information Technology
Reva University, Bengaluru, India
sunil.manvi@revainstituion.org

Abstract

Wireless sensor networks (WSNs) are usually a resource constrained networks which have limited energy, bandwidth, processing power, memory etc. These networks are now part of Internet by the name Internet of Things (IoT). To get many services from WSNs, we may need to run many applications in the sensor nodes which consumes resources. Ideally, the resources availability of all sensor nodes should be known to the sink before it requests for any further service(s) from the sensor node(s). Hence, continuous monitoring of the resources of the sensor nodes by the sink is essential. The proposed work is a framework for monitoring certain important resources of sensor network using Adaptive-Neuro Fuzzy Inference System (ANFIS) and Constrained Application Protocol (CoAP). The ANFIS is trained with these resources consumption patterns. The input to ANFIS is the resources consumption levels and the output is the resources consumed levels that needs to be sent to the sink which may be individual or combinations of resources. The trained ANFIS generates the output periodically which determines resources consumption levels that needs to be sent to the sink. Also, ANFIS continuously learns using hybrid learning algorithm (which is basically a combination of back propagation and least squares method) and updates its parameters for better results. The CoAP protocol with its observe option is used to transport the resource monitoring data from the sensor nodes to the cluster head, then from the cluster head to the sink. The sensor nodes runs coap server, the cluster head runs both coap client and server and the sink runs coap client. The performance of the proposed work is compared with LoWPAN network management protocol (LNMP) and EmNets Network Management Protocol (EMP) in terms of bandwidth and energy overheads. It is observed that proposed work performs better when compared to the existing works.

Key words: Wireless sensor networks (WSNs); Resource management; Resource Monitoring; Constrained application protocol (CoAP); ANFIS; Fuzzy inference system.

1 Introduction

The Wireless Sensor Networks as described by many researchers [1], is mainly used for monitoring the physical world. It is now connected to the Internet using new communication technologies like IEEE 802.15.4 standard, 6LoWPAN, RPL, CoAP etc. Internet of Things [2] is the larger technological term used to refer the connectivity of building automation, industrial automation, transportation, logistics, wireless sensor networks etc. with the Internet. Sensor nodes or any other real world things (embedded devices) can be connected to other sensor nodes or things in the other part of the world through global Internet infrastructure for timely collection and sharing of data as part of various physical world monitoring applications.

The design and development of WSNs is influenced by managing resources which are usually limited in WSNs, wireless radio characteristics, middleware, application specific QoS requirements, etc. Limited resources of sensor nodes which mainly include energy, bandwidth, memory space, processing power, etc. These resources of sensor nodes have to be managed effectively to provide better QoS services, reliability, greater performance and long life span of the WSNs. Resource allocation, resource mapping, resource adaptation, resource monitoring, resource discovery and selection, resource estimation, resource scheduling and resource modeling are some issues pertaining to resource management in WSNs. This paper addresses the monitoring of resources of sensor nodes.

Resource Monitoring is a systematic process of observing, tracking and recording data about resources of sensor nodes for the purposes of utilizing the services of sensor nodes and network to the maximum possible extent. Energy, bandwidth, processing capability, memory space, etc. are some of the main resources of sensor nodes. These resources are utilized by sensor node to run applications, to transmit/receive data, temporary storage of data, etc. Sink is a device which is responsible for monitoring sensor nodes resources, routinely gathers information about resources of the sensor nodes.

Resource monitoring is required in WSNs for the following reasons. (1) Gathering of information about sensor nodes of the network which can be used to make decisions about the demand for better services or more services from the sensor nodes. (2) To learn from experiences to improve in demanding services from the sensor nodes. (3) To have an accountability of resources used for different purposes and the obtained results. (4) Information gathered through monitoring which could be used to analyze, evaluate all components of the network in order to measure its effectiveness and adjust inputs where ever necessary. (5) Monitoring allows results, processes and experiences to be documented and used as a basis to steer decision making and learning processes. (6) The data acquired through monitoring is used for evaluation.

The existing IP based solutions for the resource monitoring in WSNs are based on periodic reporting about resources or use query-response techniques. The periodic reporting about the resources by the sensor nodes to the sink is irrespective of the resources consumed by different processes or applications. Information about resources are reported even when they are not consumed. This reporting is unnecessary and results in the consumption of resources (bandwidth and energy for the transport of resource monitoring data from sensor nodes to sink) which are precious in WSNs. Query processing is a two way communication which results in delay in obtaining the information about the resources and consumes extra bandwidth and energy. These are some of the problems that are addressed in this paper with solutions provided by designing Adaptive-Neuro Fuzzy Inference System (ANFIS) and using Constrained Application Protocol (CoAP) along with its observe option.

Resources to be monitored are fed as inputs to the multi-layered ANFIS [3] whose output depends on training with some input-output data pairs and subsequent learning. The consumption level of the resources and which resources need to be transported to the sink from sensor nodes makes input-output data pair. The ANFIS is trained to indicate resources consumption level to be transported when consumption exceeds 40%. If there are three resources to be monitored, then ANFIS may indicate to either transport consumption level of any one resource or combination of any two resources or all three resources depending on the resources consumed by the sensor node. The ANFIS also learns to serve better using back propagation and least squares method.

The Constrained Application Protocol [4] along with its observe option [5] and client-server architecture is used to collect the information about resources of sensor nodes. The WSN is organized into clusters where the cluster head transports the information to the sink. The cluster head is the client which collects the resources information of sensor nodes. The sensor nodes are the servers which supply their resources consumption information to the cluster head. The cluster head acts as the client while collecting resource monitored data from sensor nodes and acts as a server when it supplies the same to the sink. The client endpoint of the CoAP device registers as a client to the server(s) with its observe option. After the registration, the CoAP server with the observe option sends the resource information whenever there is a change in the resource consumption.

Our contributions are as listed below.

1. IP based resource monitoring technique is developed to efficiently collect resources consumption information from sensor nodes of the WSNs.
2. The developed resource monitoring technique uses ANFIS which is designed to enhance the efficiency of resources information collection. The ANFIS is trained to intelligently decide which resources information has to be sent to the sink.

3. The ANFIS training data is developed based on intuition and is used to optimize the membership functions of the ANFIS.
4. Finding the efficiency of the developed technique in terms of bandwidth and energy at node level, cluster level and network level.
5. Using the CoAP with observe option enhances the efficiency of developed resource monitoring technique. The combination of the coap with its observe option and ANFIS gives a superior performance (in terms of bandwidth utilization and energy consumption to transport the monitoring data) compared to LNMP (query based) and EMP (periodic reporting) based resource monitoring techniques.

The rest of the paper is organized as follows. The related work is given in section 2. The protocol stack of the sensor node is discussed in section 3. The complete work description is provided in section 4. Simulation of the work at node level, cluster level, network level and comparison with other works is described in the section 5. The section 6 concludes the paper

2 Related Work

Energy of sensor nodes is continuously monitored to avoid sensor node failure which in turn may result in the sensor network failure. Some of energy monitoring techniques are as follows. eScan [6] is a monitoring technique developed to monitor the energy levels (energy map) of sensor nodes of the network using aggregation based approach. Later predication based energy maps are developed [7] to monitor the energy levels of the sensor network. Further, monitoring the energy with low overhead is proposed [8] using the techniques such as hierarchical monitoring structure, in-network aggregation and cluster heads rotation.

Similarly monitoring techniques are developed to monitor the link quality, congestion level, bandwidth, buffer length etc. Snooping based link quality monitoring is designed [9] which listens to the channel and infers the success and loss rates. CODA [10] and ECODA [11] are receiver based congestion detection and monitoring mechanisms. All these resource monitoring mechanism deals with only one resource and that is communicated to the necessary ends (e.g. sink). Sending monitoring data using piggybacking is proposed [12]. Our work is not to improve any of these works of resource monitoring, instead, designing a framework to send consumption level of all necessary resources (like residual energy, bandwidth, buffer length, link quality) collectively and efficiently (based on some criteria) to the sink which uses these resources consumption level to monitor the sensor network services to provide services to its users.

Other related works are as follows. Some of the applications which provide sensor network management schemes in terms of controlling and monitoring [13] are BOSS, MANNA, WinMS, TinyDB, Mote-View, TP. All these are non-IP based network management systems. The LowPAN Network Management protocol (LNMP) [14] and EmNets Network Management Protocol (EMP) [15] are IP based sensor network management protocols. The LNMP uses query processing and EMP uses periodic reporting to fetch information from the sensor network. Our work uses ANFIS and CoAP with its observe option to push resource monitored data from sensor nodes to the sink.

Now sensor networks are viewed as part of Internet of things. Web services are provided with two different styles: Big Web services (SOAP) and RESTful web services. REST style web service is more suitable for sensor devices because of its simplicity. Recent work on management of sensor devices uses REST style [16] web service. This work is on complete sensor device management but not addressed networking issues. Our proposed work is not on complete device management but considers collecting resources consumption level of all sensor nodes of the network.

We have not found any literature which uses the ANFIS model in WSNs for Resource Monitoring. Recent work on Resource Mapping [17] uses ANFIS for video communication. ANFIS is used to decide which resources level has to be notified to the registered client(s) using coap with observe option.

3 Sensor node Protocol Stack

A low power, highly reliable and Internet enabled standardized protocol stack for IoT [18] is discussed by Maria Rita Palattella et al. The same protocol stack is adopted in this proposed work. Figure 1 illustrates IoT protocol stack along with protocols from physical layer to application layer.

3.1 Low Power PHY and MAC Layer – IEEE 802.15.4(2011)

The IEEE 802.15.4 - 2011 [19] defines the specifications for physical layer (PHY) and medium access layer (MAC) for low cost, low-power and low data rate wireless connectivity. The physical layer provides two services (1) The PHY data service: This service enables the reception and transmission of PHY protocol data units across the physical radio channel. (2) The PHY management service: This service provides activation and deactivation of the radio transceiver, energy detection within the channel, link quality indicator, clear channel assessment etc. Some of the PHYs defined in this standard are O-QPSK PHY, BPSK PHY, ASK PHY, CSS PHY, UWB PHY etc.

The MAC sublayer provides two services (1) The MAC data service: This service enables the reception and the transmission of MAC protocol data units

across the physical data service. (2) The MAC management service: This service provides beacon management, guaranteed time slot (GTS) management, channel access, frame validation etc.

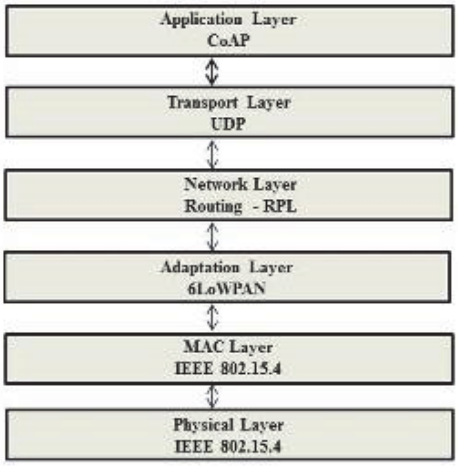


Figure 1. IoT Protocol Stack

3.2 Adaptation layer - IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs)

The 6LoWPAN, an IETF standard [20] is necessary between IPv6 layer and IEEE 802.15.4 MAC layer of the protocol stack to fit the IEEE 802.15.4 into the stack. The IEEE 802.15.4 standard has a frame size of 127-byte, with payload size of layer 2 as low as 72 bytes. The IPv6 standard has a minimum packet size of 1280 bytes, thus requiring fragmentation and reassembly. This is done by the adaptation layer 6LoWPAN.

The IPv6 and UDP headers consumes significant portion of the payload space in a single IEEE802.15.4 packet. To reduce the overhead, the header compression is provided by the 6LoWPAN

3.3 Routing – RPL

The IPv6 routing protocol for low power and lossy networks (RPL) [21] is designed considering the constrained nodes (with limited energy, memory and processing power) which supports only low data rates. It supports three basic traffic flows (1) point-to-point (2) point-to-multipoint (3) multipoint-to-point. The RPL does not rely on any particular link layer technology and designed to operate over different link layers.

The RPL is basically a distance vector routing protocol which does not require predefined topology but is able to build the topology of the network. The routes of RPL are optimized for traffic from (or to) one or more sinks (roots) of the topology. It organizes a topology of the network as a Directed Acyclic Graph (DAG). The DAG is partitioned into many Destination Oriented DAGs (DODAGs), one DODAG per sink. In multiple root DAG, the roots are joined by common backbone (transit link).

3.4 Transport Layer – UDP

The user datagram protocol (UDP) provides minimum protocol mechanism for application programs to send packets to other application programs. This protocol is transaction based and does not provide any guaranteed delivery of packet and duplicate protection. The application layer protocol is responsible for reliability (i.e. retransmission of lost packets).

3.5 Application Layer – CoAP

The Constrained Application Protocol (CoAP) is a REST style and specially designed web transfer protocol (like HTTP) for use with constrained networks and constrained nodes. It includes key concepts of the web such as built in discovery of services and resources, multicast support, Internet media types and URIs. It has two main features (1) Messaging: deals with UDP to exchange messages asynchronously between CoAP endpoints. CoAP defines four types of messages: confirmable (for reliability), Non-confirmable, acknowledgment, reset. (2) Requests and Responses: deals with the application using methods (GET, PUT, POST, DELETE) and response codes (2.xx, 4.xx, 5.xx).

The CoAP's interaction model is similar to the client-server model of HTTP. The observe extension to the CoAP offers a mechanism for a CoAP client to observe a resource on a CoAP server. With this extension of CoAP, client can retrieve a representation of the resource and request this representation to be updated by the server over a period of time as long as it is interested in the resource. The sensor node which has resources to be monitored acts as a server and sink acts as a client. The registration by the client and subsequent notifications by the server is as shown in the Figure 2. The client uses the GET request with observe option (observe = 0) of the CoAP (extended GET) for registration. After registration, the server immediately responds with current resource level with observe option. The observe option of the
CoAP when included in a response (server side), identifies the message as a notification. The server sets the value of the observe option of each notification in the increasing sequence number order. This is helpful in reordering the

notifications if a notification arrives at the client later than a newer notification from the server. The request by the client (registration) carries a self-generated token that is echoed by the server in the resulting subsequent notifications. The notification uses the 2.05 (content) as a response code which indicates that the payload in the response is a representation of the requested resource.

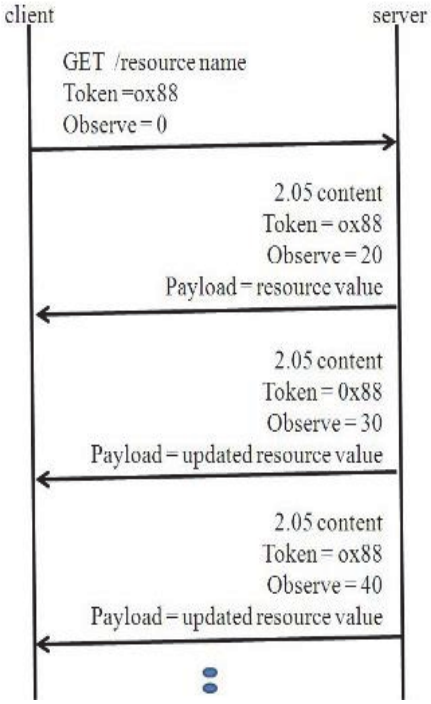


Figure 2. CoAP with observe option

4 ANFIS based Resource Monitoring using CoAP in Wireless Sensor Networks

4.1 Resources of the Sensor Networks

There are mainly two types of resources in sensor networks: physical and logical. The processor, memory, sensors etc. are important physical resources of the sensor nodes whereas energy, bandwidth, operating system, network throughput etc. are important logical resources. The extent of involvement of the processor, amount of memory occupied by the current processes, residual energy, bandwidth are the important resources of sensor nodes to be monitored in WSNs. In general, consider the resources to be monitored as R1, R2,

R3, ..., Rn. The patterns of consumption of these resources are observed over the period of time. The consumption patterns of these resources are finalized and used for monitoring of the resources.

4.2 Resource Monitoring Policy

The proposed work is the framework for the monitoring of three resources R1, R2 and R3. We assume that the consumption patterns of these resources are as shown in the Figure 3. These consumption patterns can be used for the three most important resources of sensor node: energy, memory and processing speed. These are the inputs to the ANFIS which gives the output as which resources level should be notified to the registered client based on the policy shown in the Table 1. The policy is based on the assumed resource consumption pattern and is not to notify the resources level whose consumption level is below 40% to the registered client.

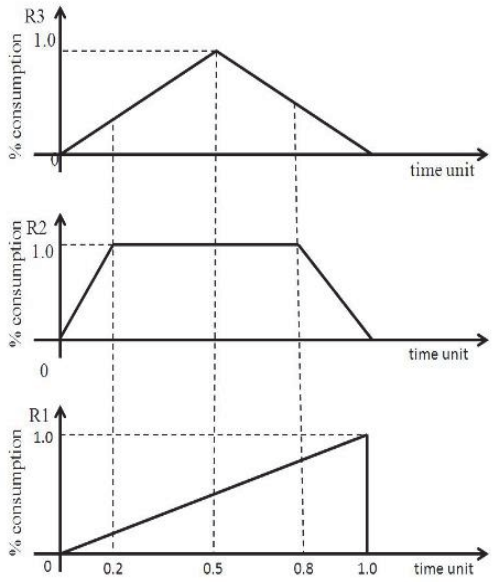


Figure 3. Resource consumption patterns

4.3 Adaptive-Neuro Fuzzy Inference System (ANFIS)

The ANFIS is a class of adaptive neural networks that are functionally same as fuzzy inference systems. The fuzzy inference system is embedded into the framework of adaptive neural networks to obtain the ANFIS. Fuzzy if-then rules and membership functions based on the zero-order Sugeno type fuzzy

inference system [22] are used to construct the ANFIS which is used to generate the required input-output data pairs of the proposed work as shown in the Table 2. The membership functions parameters are tuned using the input-output training data set and the hybrid learning algorithm (i.e. combination of back propagation and least squares method).

Table 1. Resource Monitoring Policy

Resources consumed	Resources to be notified	code
R1 < 0.4, R2 < 0.4, R3 < 0.4	NULL	0
R1 > 0.4, R2 < 0.4, R3 < 0.4	R1	1
R1 > 0.4, R2 > 0.4, R3 < 0.4	R2 R1	2
R2 > 0.4, R1 < 0.4, R3 < 0.4	R2	3
R2 > 0.4, R3 > 0.4, R1 < 0.4	R3 R2	4
R3 > 0.4, R1 < 0.4, R2 < 0.4	R3	3
R3 > 0.4, R1 > 0.4, R2 < 0.4	R3 R1	6
R1 > 0.4, R2 > 0.4, R3 > 0.4	R3 R2 R1	7

Table 2. Required input-output combination of ANFIS

Time unit	R3	R2	R1	ANFIS output	Resources to be notified
0.0	0.0	0.0	0.0	0	0
0.1	0.2	0.5	0.1	3	R2
0.2	0.4	1.0	0.2	4	R3 R2
0.3	0.6	1.0	0.3	4	R3 R2
0.4	0.8	1.0	0.4	7	R3 R2 R1
0.5	1.0	1.0	0.5	7	R3 R2 R1
0.6	0.8	1.0	0.6	7	R3 R2 R1
0.7	0.6	1.0	0.7	7	R3 R2 R1
0.8	0.4	1.0	0.8	7	R3 R2 R1
0.9	0.2	0.5	0.9	2	R2 R1
1.0	0.0	0.0	1.0	1	R1

The ANFIS structure of the proposed work consists of three inputs R3, R2 and R1, six rules, one output and zero-order Sugeno fuzzy model. The computational efficiency of Sugeno model is high and is best suited for the development of fuzzy inference system from given input-output training data set. The three inputs to the ANFIS are resource consumption patterns of R3, R2 and R1 and the single output is the code which represents which resource(s) needs to be notified to the registered client by the server. The cluster heads are the registered clients and leaf nodes are servers in a clustered tree WSNs.

List of Parameters and Notations: List of parameters and notations used in ANFIS are listed in Table 3.

Rules: The fuzzy inference system considered in the proposed work is zero order Sugeno fuzzy model which has three inputs R3, R2 and R1 and one output. Fuzzy if-then rules with R3, R2 and R1 as linguistic variables and less, more as linguistic labels is as given below.

Rule 1: if R3 is less AND R2 is less AND R1 is less then output1 = 0;
Rule 2: if R3 is less AND R2 is less AND R1 is more then output2 = 1;
Rule 3: if R3 is less AND R2 is more AND R1 is less then output3 = 3;
Rule 4: if R3 is less AND R2 is more AND R1 is more then output4 = 2;
Rule 5: if R3 is more AND R2 is more AND R1 is less then output5 = 4;
Rule 6: if R3 is more AND R2 is more AND R1 is more then output6 = 7;

where the output is represented by constant output (singular membership function).

Table 3. Parameters and notations used in ANFIS

Notation	Description
(a,b,c,d)	Premise parameter set of membership functions
less, more	Linguistic variables
$\mu_{less}(R3)$, $\mu_{more}(R3)$	Membership functions of resource R3
$\mu_{less}(R2)$, $\mu_{more}(R2)$	Membership functions of resource R2
$\mu_{less}(R1)$, $\mu_{more}(R1)$	Membership functions of resource R1
w_i	firing strength of rule i
$\overline{w_i}$	Normalized firing strength of rule i
O_i^1	i th node output of layer 1
O_i^2	i th node output of layer 2
O_i^3	i th node output of layer 3
O_i^4	i th node output of layer 4
O_i^5	i th node output of layer 5

Membership functions: The reasoning mechanism for the above Sugeno model is illustrated in the Figure 4. The membership function shown for the inputs R3, R2 and R1 with the linguistic labels less and more. The corresponding functionally equivalent ANFIS architecture which is a multi-layer network is shown in the Figure 5.

ANFIS structure: Each layer of the ANFIS performs specific task and nodes of the particular layer performs similar functions. The layers of the ANFIS are fuzzification layer, rules layer, normalization layer, defuzzification layer and output layer. The square represents a adaptive node whereas the circle indicates a fixed node. The output signals from nodes of a layer are fed as inputs to the next layer as shown in the Figure 5.

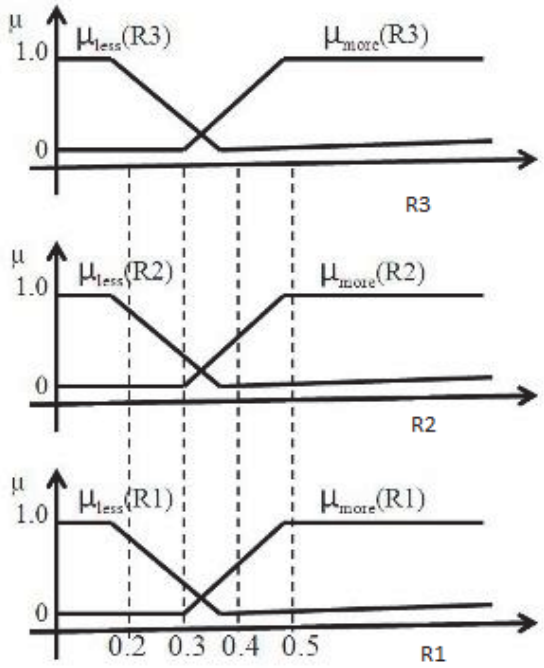


Figure 4. Membership functions

Layer 1: This layer is called the fuzzification layer which fuzzifies the inputs R3, R2 and R1. Nodes of this layer are adaptive nodes with node function

$O_1^1 = \mu_{less}$ (R3)

$O_2^1 = \mu_{more}$ (R3)

$O_3^1 = \mu_{less}$ (R2)

$O_4^1 = \mu_{more}$ (R2)

$O_5^1 = \mu_{less}$ (R1)

$O_6^1 = \mu_{more}$ (R1)

where R3, R2 and R1 are inputs to the nodes at this layer and less/more is the linguistic label associated with the node function. In general, O_i^1 (the function of the i^{th} node) is the membership function of less/more and it specifies the degree to which the given input (R3/R2/R1) satisfies the quantifier less/more. We choose the membership function to be trapezoidal in shape

$$trap(x:a,b,c,d) = max(min((x-a) / (b-a),1,(d-x) / (d-c)),0) \quad (1)$$

where {a,b,c,d} is the premise parameter set (with a < b <= c < d) which determines the x coordinates of the four corners of the trapezoidal membership

functions. The change in the values of these parameters varies the trapezoidal function accordingly and produce various forms of the membership functions.

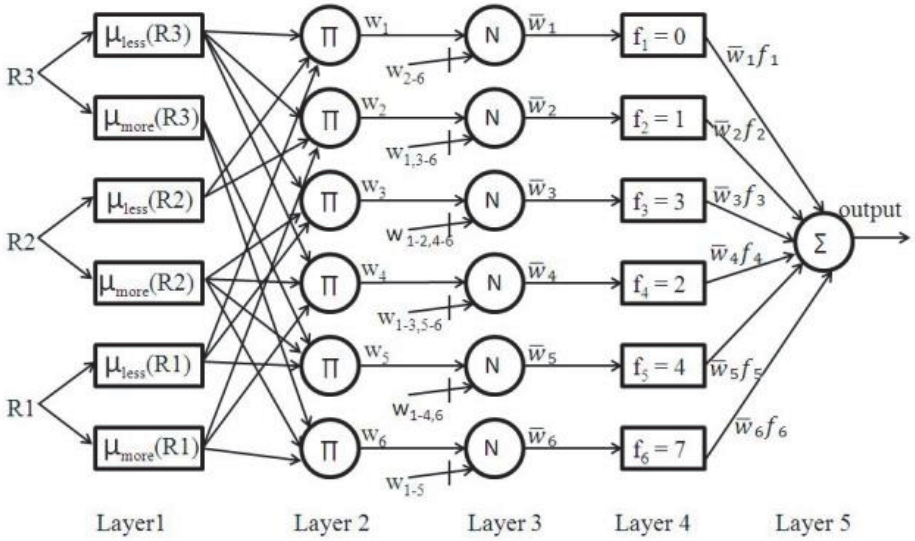


Figure 5. ANFIS structure

Layer 2: This layer is called the rules layer. All nodes of this layer are fixed nodes (represented by π) and performs similar functions. The inputs of each node are connected by the intersection operator (fuzzy AND) as shown in the if-then rules. The output of each node is the product of all its incoming signals. Six nodes are used to implement six rules whose outputs are as follows.

$O_1^2 = \mu_{less}(R3) \times \mu_{less}(R2) \times \mu_{less}(R1)$

$O_2^2 = \mu_{less}(R3) \times \mu_{less}(R2) \times \mu_{more}(R1)$

$O_3^2 = \mu_{less}(R3) \times \mu_{more}(R2) \times \mu_{less}(R1)$

$O_4^2 = \mu_{less}(R3) \times \mu_{more}(R2) \times \mu_{more}(R1)$

$O_5^2 = \mu_{more}(R3) \times \mu_{more}(R2) \times \mu_{less}(R1)$

$O_6^2 = \mu_{more}(R3) \times \mu_{more}(R2) \times \mu_{more}(R1)$

$\mu_{less}(R3)$, $\mu_{more}(R3)$, $\mu_{less}(R2)$, $\mu_{more}(R2)$, $\mu_{less}(R1)$, and $\mu_{more}(R1)$ are fuzzified inputs to the nodes of the rules layer as shown in the Figure 5. The output of each node represents the firing strength of the rule.

Layer 3: Nodes in this layer are fixed nodes and they are represented by N. The nodes output of this layer are called the normalized firing strengths. The i^{th} node function is to calculate the ratio of firing strength of the rule i to the sum of firing strengths of all rules.

$$\overline{w_i} = \frac{w_i}{\sum_{k=1}^{6} w_k} \tag{2}$$

where i = 1,2,3,4,5,6

Layer 4: Nodes of this layer are adaptive nodes with a node function

$$O_i^4 = \overline{w_i} * f_i = \overline{w_i} * \text{constant} \tag{3}$$

where $\overline{w}_i$ is the normalized firing strength.

Layer 5: The only node in this layer is a fixed node (represented by ∑) which computes the overall output as the sum of all the incoming signals.

$$O_i^5 = \sum_{i=1}^{6} \overline{w}_i f_i = \frac{\sum_{i=1}^{6} w_i f_i}{\sum_{i=1}^{6} w_i} \tag{4}$$

The learning mechanism has to tune only the premise parameters to fine tune the membership functions. In the proposed work, the ANFIS is modelled as zero-order Sugeno fuzzy inference system and there is no consequent parameter set.

ANFIS input-output training data: The range of three inputs R3, R2 and R1 is from 0.0 to 1.0 which corresponds to 0% to 100% of the resource consumed and the output code is from 0 to 7 which corresponds to the resources to be notified to registered client like cluster head.The ANFIS input-output training data set is prepared as follows. The resources R3 and R2 are fixed at 0.0 and R1 is varied from 0.0 to 1.0 in steps of 0.1. Then R2 is changed (keeping R3 same) to 0.1 and R1 is varied again from 0.0 to 1.0 in steps of 0.1. This is repeated by changing R2 (keeping R3 same) in steps of 0.1 upto 1.0. Now R3 is changed to 0.1 (with R2 = R1 = 0) and the above procedure is repeated. The R3 is changed upto 1.0 in steps of 0.1 and the procedure is repeated for every incremental value of R3. The output code for the combination of R3, R2 and R1 consumption levels is in accordance with the Table 2. Part of the training data with R3 = 0.3 is shown in the Table 4.

4.4 Pushing monitoring data from sensor nodes to sink using CoAP's observe option

This Resource Monitoring application uses CoAP's observe option to push the monitoring data from sensor nodes to sink. This is done at three levels. (1) Sensor node level (2) Cluster level (3) Network level. The clustered tree network structure is considered in the proposed work and is as shown in the Figure 6. The tree like structure is considered for simplicity, although the IPv6

routing protocol allows for each sensor node to have multiple parents when the node's connectivity supports it.

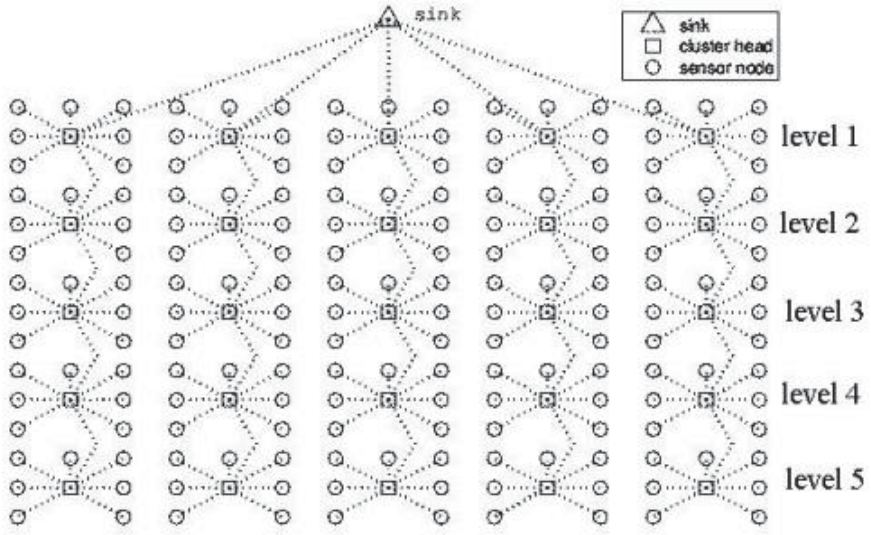


Figure 6. Clustered Tree Network

Sensor node level: At this level, the single sensor node which serves as a server (has resource monitored data) connected with the sink which is a client. The client registers to the server its interest in resources R3, R2 and R1 using an extended GET request (i.e. with observe option). The server sends notification to the client as indicated in Table 2 and resource consumption patterns in Figure 3.

Cluster level: The cluster head is a special node which is rich in resources and capabilities (e.g. routing) compared to sensor nodes. Sensor nodes are connected to cluster head to form star network inside the cluster. The cluster head (client) registers its interest in resources R1, R2 and R3 by initiating an extended GET request (i.e. with observe option) to all sensor nodes (servers)

Table 4. ANFIS Training Data

R3	R2	R1	o/p	R3	R2	R1	o/p	R3	R2	R1	o/p
0.3	0.0	0.0	0	0.3	0.3	0.7	1	0.3	0.7	0.3	3
0.3	0.0	0.1	0	0.3	0.3	0.8	1	0.3	0.7	0.4	2
0.3	0.0	0.2	0	0.3	0.3	0.9	1	0.3	0.7	0.5	2
0.3	0.0	0.3	0	0.3	0.3	1.0	1	0.3	0.7	0.6	2
0.3	0.0	0.4	1	0.3	0.4	0.0	3	0.3	0.7	0.7	2
0.3	0.0	0.5	1	0.3	0.4	0.1	3	0.3	0.7	0.8	2
0.3	0.0	0.6	1	0.3	0.4	0.2	3	0.3	0.7	0.9	2
0.3	0.0	0.7	1	0.3	0.4	0.3	3	0.3	0.7	1.0	2
0.3	0.0	0.8	1	0.3	0.4	0.4	2	0.3	0.8	0.0	3
0.3	0.0	0.9	1	0.3	0.4	0.5	2	0.3	0.8	0.1	3
0.3	0.0	1.0	1	0.3	0.4	0.6	2	0.3	0.8	0.2	3
0.3	0.1	0.0	0	0.3	0.4	0.7	2	0.3	0.8	0.3	3
0.3	0.1	0.1	0	0.3	0.4	0.8	2	0.3	0.8	0.4	2
0.3	0.1	0.2	0	0.3	0.4	0.9	2	0.3	0.8	0.5	2
0.3	0.1	0.3	0	0.3	0.4	1.0	2	0.3	0.8	0.6	2
0.3	0.1	0.4	1	0.3	0.5	0.0	3	0.3	0.8	0.7	2
0.3	0.1	0.5	1	0.3	0.5	0.1	3	0.3	0.8	0.8	2
0.3	0.1	0.6	1	0.3	0.5	0.2	3	0.3	0.8	0.9	2
0.3	0.1	0.7	1	0.3	0.5	0.3	3	0.3	0.8	1.0	2
0.3	0.1	0.8	1	0.3	0.5	0.4	2	0.3	0.9	0.0	3
0.3	0.1	0.9	1	0.3	0.5	0.5	2	0.3	0.9	0.1	3
0.3	0.1	1.0	1	0.3	0.5	0.6	2	0.3	0.9	0.2	3
0.3	0.2	0.0	0	0.3	0.5	0.7	2	0.3	0.9	0.3	3
0.3	0.2	0.1	0	0.3	0.5	0.8	2	0.3	0.9	0.4	2
0.3	0.2	0.2	0	0.3	0.5	0.9	2	0.3	0.9	0.5	2
0.3	0.2	0.3	0	0.3	0.5	1.0	2	0.3	0.9	0.6	2
0.3	0.2	0.4	1	0.3	0.6	0.0	3	0.3	0.9	0.7	2
0.3	0.2	0.5	1	0.3	0.6	0.1	3	0.3	0.9	0.8	2
0.3	0.2	0.6	1	0.3	0.6	0.2	3	0.3	0.9	0.9	2
0.3	0.2	0.7	1	0.3	0.6	0.3	3	0.3	0.9	1.0	2
0.3	0.2	0.8	1	0.3	0.6	0.4	2	0.3	1.0	0.0	3
0.3	0.2	0.9	1	0.3	0.6	0.5	2	0.3	1.0	0.1	3
0.3	0.2	1.0	1	0.3	0.6	0.6	2	0.3	1.0	0.2	3
0.3	0.3	0.0	0	0.3	0.6	0.7	2	0.3	1.0	0.3	3
0.3	0.3	0.1	0	0.3	0.6	0.8	2	0.3	1.0	0.4	2
0.3	0.3	0.2	0	0.3	0.6	0.9	2	0.3	1.0	0.5	2
0.3	0.3	0.3	0	0.3	0.6	1.0	2	0.3	1.0	0.6	2
0.3	0.3	0.4	1	0.3	0.7	0.0	3	0.3	1.0	0.7	2
0.3	0.3	0.5	1	0.3	0.7	0.1	3	0.3	1.0	0.8	2
0.3	0.3	0.6	1	0.3	0.7	0.2	3	0.3	1.0	0.9	2
Contd.				**Contd.**				0.3	1.0	1.0	2

in the cluster. Sink (client) registers its interest in resources R1, R2 and R3 of all sensor nodes of cluster by initiating a extended GET request to the cluster head (acts as server to the sink).

Network level: In the hierarchical tree structure shown in the Figure 6, the sink is at depth 0, and clusters are arranged at different depths 1, 2, 3, ..., n. The cluster head acts as client for sensor nodes inside its cluster and server for cluster head of upper cluster head and acts as client of lower cluster head.

5 Simulation, Results and Analysis

5.1 Simulation Environment

Matlab 7 (R2010a) and its fuzzy logic tool box is used to simulate the ANFIS to generate the membership functions and rules from the training data. We have reduced the ANFIS generated rules from eight to six. The GUI tool anfisedit is used in the simulation process. ANFIS is trained using hybrid learning algorithm.

The sensor node, cluster head, and network environment is simulated using the Java programming language and CoAP with observe option used for transporting resources information is implemented using the Californium (CoAP framework).

5.2 Node level

Simulation: The sensor node is equipped with ANFIS model. The ANFIS model consists of 25 nodes, three inputs (R3, R2 and R1), six membership functions, six rules and one output. It is trained with 1,331 pairs of input and output training data (Table 4). The parameters (a,b,c,d) of trapezoidal membership functions (two membership functions for each input) $\mu_{less}(R3)$, $\mu_{more}(R3)$, $\mu_{less}(R2)$, $\mu_{more}(R2)$, $\mu_{less}(R1)$ and $\mu_{more}(R1)$ are (-0.7, -0.3, 0.14, 0.39), (0.3, 0.54, 1.3, 1.7), (-0.7, -0.3, 0.14, 0.39), (0.3, 0.54, 1.3, 1.7), (-0.7, -0.3, 0.14, 0.39) and (0.3, 0.54, 1.3, 1.7) respectively. The only output of the ANFIS (Table 2) is the coded output which indicates which resources needs to be pushed (notified) to the client.

The sensor node which has resources to be monitored acts as a server and the sink acts as the client. The registration by the client and subsequent notifications by the server is as shown in the Figure 7. The client uses the GET request with observe option of the CoAP (extended GET) for registration (observe = 0). After registration how communication happens between client

and server using token, observe option, response code and payload is already explained under protocol stack. The Figure 7 is simplified version of Figure 2.

All the assumed resources consumption patterns are as shown in the Figure 3. After registration from the client, the server sends the notifications (which are based on Table 2) for every 0.1 time unit up to 1.0 time unit as shown in the Table 2. The detailed notifications for one complete cycle of resources consumption patterns are shown in the Figure 7. The payload of the notification is the resources consumption level which may be just any one resource level (R3 or R2 or R1) or combination of any two or all three resources level as shown in the notifications.

The frequency of resource monitoring data movement from the sensor nodes to the sink depends on the magnitude of the resources of the sensor node and resources consumption patterns. The time unit may vary from network to network.

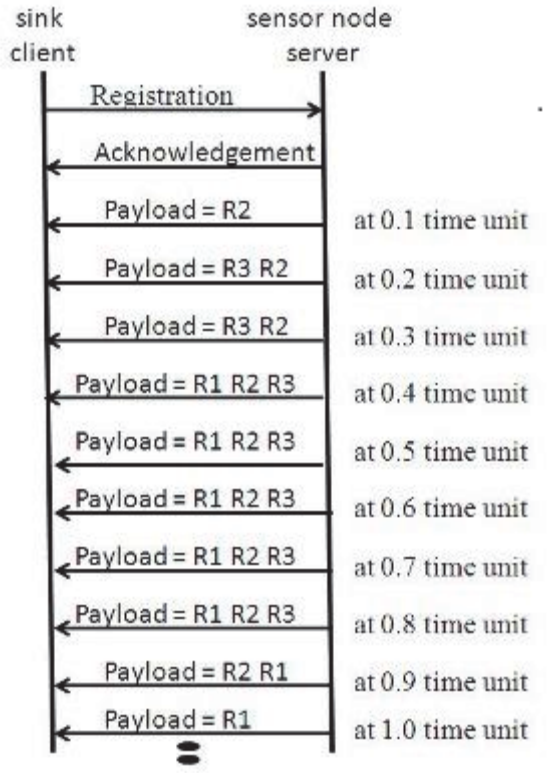


Figure 7. Interaction between node and sink

Results and Analysis: Resources consumption level cannot be sent in % as the network may have heterogeneous sensor nodes which have different size

of memory, processor with different speed and capabilities etc. Let us assume that each resource consumption level indication needs 4 bytes. Then for three resources, 12 bytes required. These resources consumption level is notified to the client for every 0.1 time unit. The complete cycle of one time unit then needs 12 x 10 = 120 bytes of transmission to client. Instead of notifying all three resources to the client (EMP, periodic reporting), the ANFIS decides which resources level needs to be notified to the client based data provided in the Table 2. For the assumed resources consumption pattern (Figure 3), the resources need to be notified is already shown in the Figure 7. The number of bytes to be transmitted for one cycle is graphically shown in the Figure 8(a). Total number of bytes needs to be transmitted is 92 bytes which is 23% less compared to notifying all resources consumption level. This clearly indicates a 23% less consumption of energy by the sensor node and 23% less bandwidth required for the transmission of resource monitoring data. Here the saving of sensor node energy and required bandwidth is only indicative and vary based on resource consumption pattern.

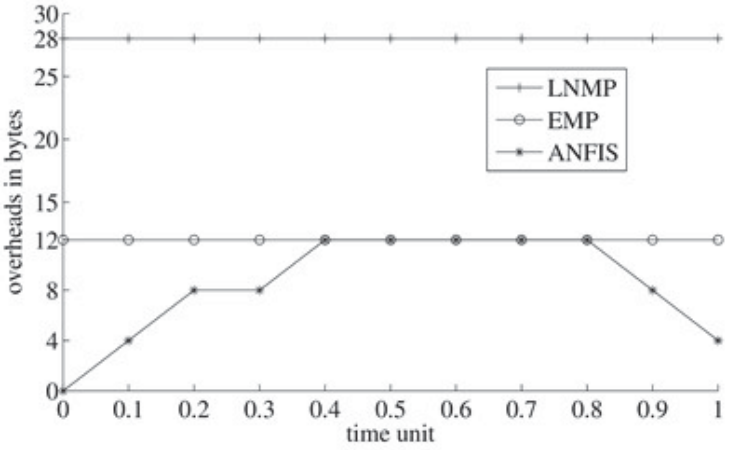
The query processing used in LNMP demands a query request from the client for every fetch of resources information. The CoAP protocol consumes 16 bytes for (ignoring the headers from other layers) a query request. We need to query the server 10 times for one complete cycle. A one complete cycle demand the transmission of 12x10 + 16x10 = 280 bytes (includes both request and response) which is 67% higher compared to our work. Hence, the saving of 67% energy and 67% less bandwidth requirement for the transport of resource monitoring data at the node level when ANFIS and CoAP is used.

The radio model of IEEE 802.15.4 standard approximately consumes 200nJ/bit (based on products survey) and 1600nJ/byte for transmission or reception. Figure 8(b) shows the comparison of energy consumption in LNMP, EMP and ANFIS when used for resource monitoring for 10 time units. The graph clearly shows the energy saving (in sensor node with ANFIS and CoAP) of 23% when compared to EMP and 67% when compared to LNMP.

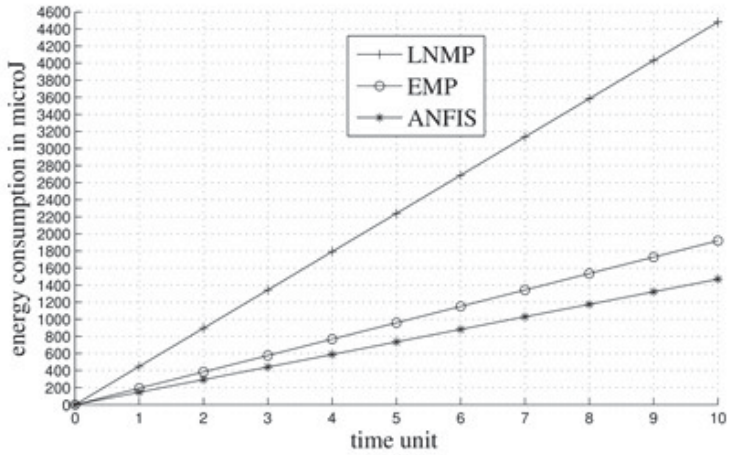
5.3 Cluster level

Simulation: Let us consider the cluster of sensor nodes with one special node which functions as cluster head. All sensor nodes are similar and equipped with the ANFIS as described in the node level section. The resources consumption in all sensor nodes is assumed to be same. In the Figure 6 shown, there are seven sensor nodes which are wirelessly connected to the cluster head which in turn wirelessly connected to the sink. The sink which needs resource monitoring data of all the sensor nodes registers itself as a client to the cluster head which acts as server capable of feeding the monitoring data of all sensor nodes to the sink. The cluster head registers as client to all sensor

nodes separately for requesting monitoring data. The process of registration and resource monitoring data movement from sensor nodes to cluster head and then from cluster head to sink is shown in the Figure 9. The details of process of registration by the client and notifications from the server are exactly same as details presented in the node level section.



(a) Bandwidth



(b) Energy

Figure 8. Performance at node level

The notifications from the sensor node (servers) at 0.1 unit time is collected by the cluster head (acts as client) and all these notifications are sent by cluster head (acts as server) to sink (client) in a single packet. Subsequent notifications from the sensor nodes at 0.2, 0.3, 0.4, ..., 0.9 and 1.0 time unit is collected by the cluster head (client) and sent to the sink in sequence as shown in the Figure 9.

Results and Analysis: The total number of bytes required to send all resources monitoring data (R3, R2, R1) of all sensor nodes in a cluster from the cluster head to the sink is 7 sensors x 3 resources x 4 bytes/resource x 10 times/cycle = 840 bytes. The sensor nodes equipped with ANFIS are capable of deciding which resources consumption level needs to be sent to the cluster head. The Figure 10(a) indicates how many bytes needs to be transmitted for one complete cycle of consumption pattern of resources R3, R2, R1. At 0.1 time unit, only R2 of all sensor nodes is sent to cluster head which is 7 sensor nodes x 1 resource x 4 bytes = 28 bytes. For one complete cycle of resources consumption pattern, the total number of bytes received by the cluster head from all sensor nodes together is 644 bytes which is sent to sink.

This is approximately 23% less compared to 840 bytes (EMP, periodic reporting) which indicates 23% saving in energy consumption and bandwidth requirement for transporting monitoring data from cluster head to the sink.

The resources information of sensor nodes reaches the sink in two hops. There is a 23% saving of energy and bandwidth for each hop.

The query processing used in LNMP demands the transmission of (84 bytes x 10 times + 16 bytes x 10 times x 7 sensors) = 1960 bytes which is 67% more compare to our work. Hence, the saving of 67% energy and bandwidth for each hop.

Figure 10(b) shows the comparison of energy consumption in LNMP, EMP and ANFIS when used for resource monitoring for 10 time units and one hop (cluster head to sink). The graph clearly shows the energy saving of 23\% when compared to EMP and 67\% when compared to LNMP.

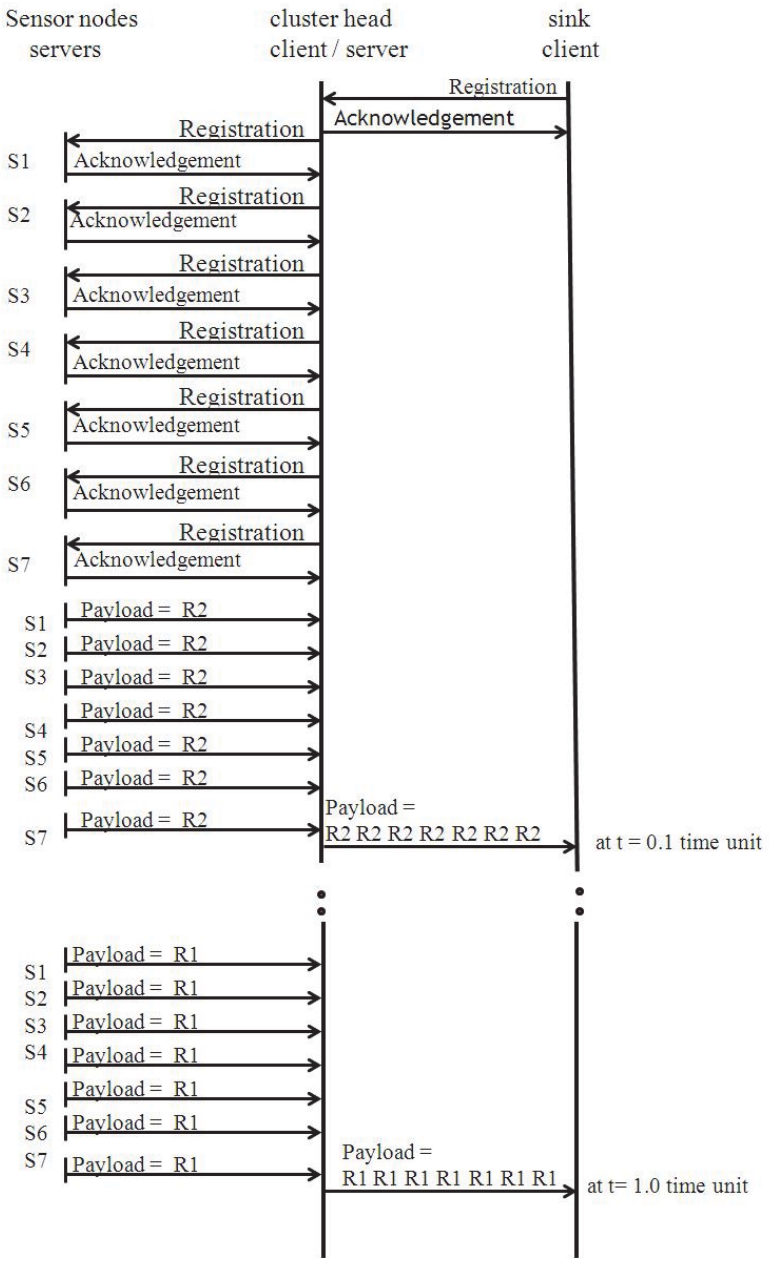
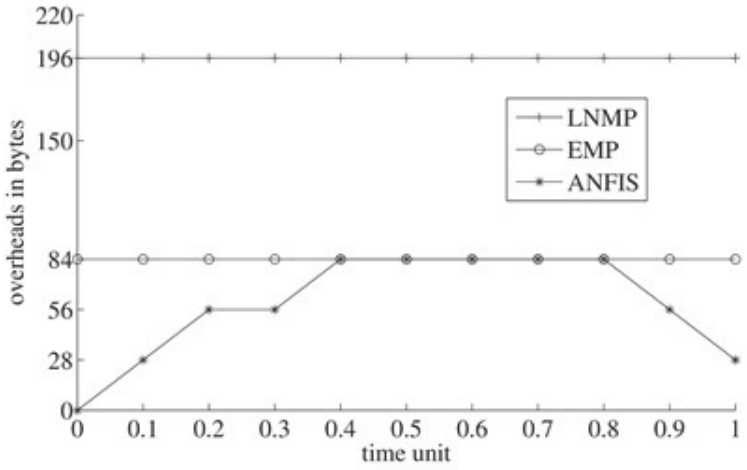
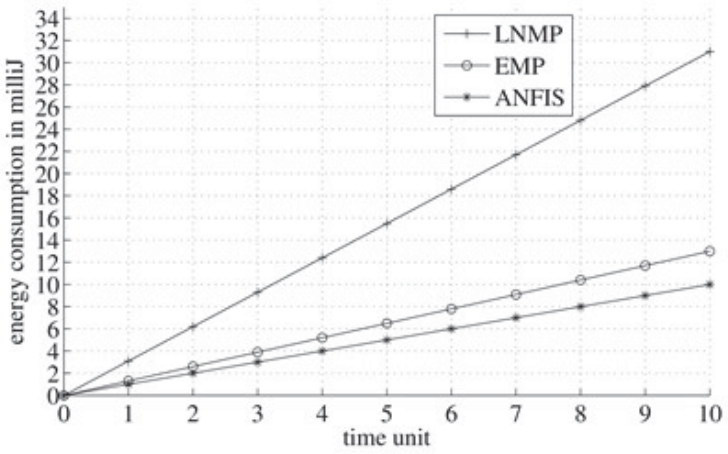


Figure 9. Interaction between sensor nodes, cluster head and sink



(a) Bandwidth



(b) Energy

Figure 10. Performance at cluster level

5.4 Network level

Analysis: By using the ANFIS, the transmission of resource monitoring data from sensor node to cluster head is reduced from 120 bytes/time unit (EMP, periodic reporting) into 92 bytes/time unit. The network of n sensor nodes with ANFIS in each node reduces the need of transmitting 120 x n bytes into 92 x n bytes which is 23% saving of energy and bandwidth requirement for transporting monitoring data from sensor nodes to the clustered head.

Let us consider the clustered tree network with equal number of sensor nodes in each cluster for analysis. The number of bytes saved is 28 bytes x number of sensor nodes in the cluster x number of branches from the root (sink) x depth of tree (sink at depth 0). For 175 sensor nodes clustered tree network with 7 sensor nodes in each cluster, with five branches and five levels as shown in the Figure 6, the number of bytes saved is 28 x 7 x 5 x 5 = 4900 bytes/time unit which is 23% saving compared to 21000 bytes/time unit (EMP, periodic reporting) as shown in the Figure 11(a). Again there is a saving of 23% of energy and bandwidth while transporting each cluster monitoring data from cluster head to sink. The monitoring data movement from cluster head to sink takes more than one hop if the cluster is at depth greater than one. There is a saving of 23% of energy and bandwidth for each hop of data movement from cluster head.

The query processing for fetching resources information using LNMP demands 280 bytes/time unit as compared to 92 bytes/time unit required by ANFIS. This is about 67% less data movement from sensor node to cluster head and saves 67% energy and bandwidth. For the above mentioned clustered tree network, it is the saving of 188 x 7 x 5 x 5 = 32,900 bytes/time unit which is 67% saving compared to 49,000 bytes/time unit (LNMP, query processing) as shown in the Figure 11(a). Hence, there is saving of 67% of energy and bandwidth for each hop of the data movement from cluster head.

Figure 11(b) shows the comparison of energy consumption in LNMP, EMP and ANFIS when used for resource monitoring for 10 time units. The graph clearly shows the energy saving in sensor node with ANFIS and CoAP (257 mJ for 10 time units) of 23% when compared to EMP (336 mJ for 10 time units) and 67% when compared to LNMP (784 mJ for 10 time units).

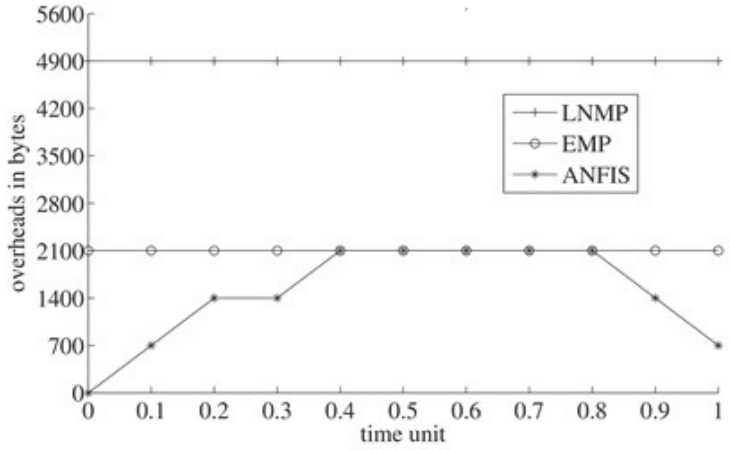
If there are two different resources consumption patterns (R3, R2, R1 for 'a' nodes and R6, R5, R4 for 'b' nodes where a + b = 7) in each cluster, then number bytes saved from transmission is different for different resource consumption pattern (say p \& q). Then bytes saved from transmission is (p x a x 5 x 5) + (q x b x 5 x 5).

In general, for a sensor network of n nodes with different resources consumption patterns for different sensor nodes

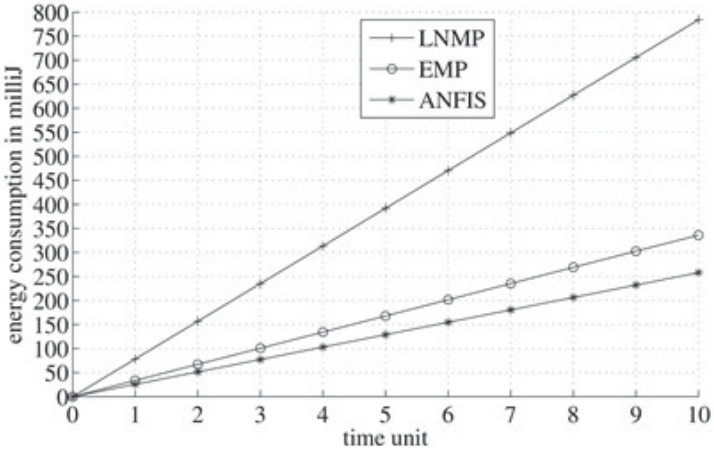
Total number of bytes saved from transmission (nodes to cluster head)

$$= \sum_{i=1}^{n} a_i$$

where a_i - bytes saved from transmission for i^{th} sensor node per hop, i = 1, 2, 3, ..., n-1,n;



(a) Bandwidth



(b) Energy

Figure 11. Performance at network level

The saving of energy in all the above cases is at the transmitting end as well as at the receiving end (receiving also consumes energy).

6 Conclusions

WSNs are becoming part of Internet of Things and getting merged into mainstream Internet. In the proposed work for resource monitoring of WSNs, sensor nodes and network are considered as Internet of Things. Each and every sensor node is simulated as web server, cluster head as both web server and web client, and sink as web client. Data related to monitoring of resources of

sensor node (processing speed, memory, energy, bandwidth) are pushed to the client (cluster head) using the observe option of the CoAP protocol. The sensor nodes are equipped with the ANFIS which decides which resources are to be pushed to the client. Use of ANFIS in sensor nodes reduces resource monitoring data size by 23\% per hop when compared to EMP (periodic reporting) and 67\% per hop when compared to LNMP (query processing). This results in the saving of energy and bandwidth requirement (in sensor node with ANFIS) by 23\% when compared to EMP and 67\% when compared to LNMP. The simulation is conducted at two levels: (1) node level: A sensor node which is equipped with ANFIS notifies resource monitoring data to sink which is a client. (2) Cluster level: All sensor nodes (web servers) of the cluster are equipped with the ANFIS and notify resource monitoring data to the cluster head (web client) which in turn notifies (now acts as web server) the same to the sink (web client). Effect of using ANFIS in senor node for resource monitoring in clustered tree network is analyzed.

References

1. Ian. F. Akyildiz, Weillan Su, Yogesh Sankarasubramaniam and Erdal Cayirci, 2002, *A Survey on Sensor Networks*, IEEE Communication Magazine Vol.40, No.8, pp.102-114.
2. Zack Shelby, 2010. *Embedded Web Services*, IEEE Wireless Communications, Vol.17, No.6, pp.52-57.
3. Jyh-shing Roger Jang, 1993, *ANFIS: Adaptive network base Fuzzy Inference System*, IEEE Transactions on Systems, Man and Cybernetics Vol.23, No.03, pp.665-685
4. Shelby Z., Hartke K. and Bormann C., 2014, *The IETF's, The Constrained Application Protocol*, https://datatracker.ietf.org/doc/rfc7252/
5. Hartke K., 2015, *Observing Resources in the Constrained Application Protocol (CoAP)*, http://www.rfc-editor.org/info/rfc7641/
6. Yonggang Jerry Zhao, Ramesh Govindhan and Deborah Estrin, 2002, *Residual Energy scan for Monitoring Sensor Networks*, https://escholarship.org/uc/item/2st0t8cf
7. Raquel A. F. Mini, Max do Val Machado, Antonio A. F. Loureiro and Badri Nath, 2005, *Prediction - based energy map for wireless sensor networks*, Ad Hoc Networks, Vol 3, No. 2, pp. 235-253
8. Edward Chan and Song Han, 2009*, Energy Efficient Residual Energy Monitoring in Wireless Sensor Networks*, International Journal of Distributed Sensor Networks, Vol 5, pp. 748-770
9. Alec Woo, Terence Tong and David Culler, 2003, *Taming the Underlying Challenges of Reliable Multihop Routing in Sensor Networks,* SenSys' 03, November 5-7, 2003, Los Angeles, California, USA.

10. Chieh-Yih, Shane B. Eisenman and Andrew T. Campbell, 2003, *CO-DA:Congestion Detection and Avoidance in Sensor Networks*, SenSys' 03, November 5-7, 2003, Los Angeles, California, USA.
11. Li Qiang Tao and Feng Qi Yu, 2010, *ECODA:enhanced congestion detection and avoidance for multiple class of traffic in sensor networks*, IEEE Transaction on Consumer Electronics, Vol 56, No. 3, pp.1387-1394
12. Falko Dressler and Dominik Neuner, 2013, *Energy-Efficient Monitoring of Distributed System Resources for Self-Organizing Sensor Networks*, IEEE Topical conference on wireless sensors networks (WiSNET), Jan 20-23, 2013 Austin, TX, PP.145-147
13. Winnie Louis Lee, Amitava Datta and Rachel Cardell-Oliver, 2007, *Network Management in Wireless Sensor Networks,* https://cpn.unl.edu/
14. Hamid Mukhtar, Kim Kang-Myo, Shafique Ahmad Chaudhry, Ali Hammad Akbar, Kim Ki-Hyung, Seung-Wha Yoo, 2008, *LNMP-Management architecture of IPv6 based low-power wireless Personal Area Networks (6LoWPAN)*, IEEE Network operations and Management Symposium, April 7-11 2008, Salvador Bahia, pp.417-424
15. Shafique Ahmad Chaudhry, Weiping Song, Muhammad Habeeb Vulla, Cormac Sreenan, 2011, *EMP:A Protocol for IP Based Wireless Sensor Networks Management*, Journal of Ubiquitous Systems and Pervasive Networks, Vol. 2, No. 1, pp. 15-22.
16. Zhengguo Sheng, Hao Wang, Changchuan Yin, Xiping Hu, Shusen Yang, Victor C. M. Leung, 2015, *Lightweight Management of Resource Constrained Sensor Devices in Internet of Things*, IEEE Internet of Things Journal, Vol. 2, No.5, pp. 402-411.
17. Nagesha and Sunilkumar S. Manvi, 2016, *ANFIS based Resource Mapping for Query Processing in Wireless Multimedia Sensor Networks,* Journal of Intelligent Systems, accepted, http://www.degruyter.com/printahead/j/jisys
18. Maria Rita Palattella, Nicola Accettura, Xavier Vilajosana, Thomas Watteyne, Luigi Alfredo Grieco, Gennaro Boggia, Mischa Dohler, 2013, *Standardized protocol stack for the Internet of (Important) Things*, IEEE Communication Surveys and Tutorials, Vol. 15, No. 3, pp.1389-1406.
19. Wireless personal area network (WPAN) working group, 2011, *Low-Rate Wireless Personal Area Networks (LR-WPANs),* http://standards.ieee.org.
20. Kushalnagar N., Montenegro G. and Schumacher C., 2007, *IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs): Overview, Assumptions, problem Statement, and Goals*, https://datatracker.ietf.org/doc/rfc4919/
21. Winter,T., Thubert,P., Brandt,A., Hui,J., Kelsey,R., Levis,P., Pister,K., Struik,R., JP.Vasseur and Alexander,R., 2012, *RPL:IPv6 Routing Protocol for Low-Power and Lossy Networks,* https://datatracker.ietf.org/doc/rfc6550/
22. Tomohiro Takagi and Michio Sugeno, 1985, *Fuzzy Identification of Systems and its Application to Modelling and Contro*l, IEEE Transactions on Systems, Man and Cybernetics, Vol. 15, pp.116-132