View metadata, citation and similar papers at core.ac.uk
brought to you by CORE
provided by Assam Don Bosco University Journals

Simulation based Evaluation of Attribute Aware Scheduling in Heterogeneous Wireless Sensor Networks

Dr. Tejas Vasavada[1], Dr. Sanjay Srivastava[2]

[1]Assitant Professor,
Lukhdhirji Engineering College, Morbi, Gujarat, India.
tejasmukeshvasavada@gmail.com

[2]Professor
Dhiirubhai Ambani Institute of Information & Communication Technology, Gandhinagar, Gujarat, India.
sanjay_srivastava@daiict.ac.in

Abstract*: *In many applications of sensor networks, it is required to sense multiple physical parameters of the same region. So multiple different types of sensors are deployed. Such networks are known as heterogeneous networks. In tree -based heterogeneous networks, complete aggregation is not possible at every node. The reason is that parent and child node may be of different types. The term Attribute is used to refer to type of packet. When objective is to maximize aggregation, parent selection should be done such that packet sent by given node should be aggregated as soon as possible in its path towards the sink. This approach would result in reduction in schedule length of the tree. Such an algorithm is known as Attribute Aware Scheduling Algorithm. In this work, one such algorithm is evaluated through simulations. It is found that Attribute Aware Scheduling results in better aggregation, smaller schedule length, and reduction in energy consumption. The reduction in schedule length means smaller latency and reduction in energy consumption means extended network lifetime.

Keywords*: *Scheduling, Tree Formation, Heterogeneous Networks, Distributed Algorithm, Sensors
(Article history: Received: 13th Nov 2019 and accepted 31st July 2020)

I. INTRODUCTION

The sensor nodes are used to sense different physical quantities like temperature, pressure, humidity, solar radiation and many others. When the network has all the nodes of the same type, the network is known as Homogeneous Network. In many applications, it is required that network should sense multiple parameters. For example, we wish to sense humidity, temperature and solar radiation of the same place. So, different types of sensor nodes have to be deployed. Such a network is known as Heterogeneous Network ([1]).

When tree is formed, every node selects one node as parent. The sink is the root of the tree. Thus a node sends its packets to the sink through parent node. Every node is assigned a time slot. The child node transmits the parent during the assigned time-slot.

Transmission of packets from sensor nodes to sink nodes is defined as convergecast.[2]. If parent node combines all the incoming packets and its own packet into a single outgoing packet, the convergecast is known as aggregated convergecast. If individual packets are forwarded, it is known as raw convergecast.

When the given node wants to select a parent node from its neighbors, it can use different criteria: (i) selecting the node which is nearest to the sink (ii) selecting the node with minimum number of unscheduled neighbors (iii) selecting the nearest node as parent (iv) selecting the node with highest residual energy as parent.

In case of heterogeneous networks, none of the above mentioned criteria is suitable. Every packet should be sent/forwarded to the parent where it can be aggregated. Based on this idea, a parent selection algorithm for heterogeneous networks is proposed in [9]. It is an extension of DICA[6]. In [9], the algorithm is evaluated through simulations using Network Simulator 2 (NS-2.35).

There are some possible improvements in the work done in [9]: (a) In simulation setup, grid topology is used. But protocol evaluation should be done considering random node deployment so that results are unbiased. (b) In [9], simulation results of control overhead, energy consumption during control phase and energy consumption during data phase are not consistent.

In this work, the algorithm proposed in [9] is evaluated using random node deployment as opposed to grid

topology used in [9]. As simulations are carried for different scenarios of random node deployment, all the simulations results are consistent. Thus the main contribution of this paper is to present performance of attribute-aware scheduling in random node deployment.

II. Related Work

There are many papers on scheduling and tree formation. In this section, some important ones are summarized.

In [3],[4],[5],[6],[7] and [8] different distributed scheduling algorithms are proposed. The core idea behind all the algorithms is explained in next few paragraphs.

Every node selects such a transmission slot such that no neighbor node is receiving in that slot. That is, the node's transmission should not create collision at neighboring nodes. To select collision-free schedule, node needs to know the transmission and reception slots of neighboring nodes. This involves message exchange among the neighbors. These extra messages are also known as control overhead.

In [6], a joint approach towards scheduling & tree formation is presented. It is termed as DICA (Distributed algorithm for Integrated tree Construction and data Aggregation). It is explained that slot selection and parent selection should be done at the same time by every node. The advantage is that node could select a parent to whom it could transmit in the lowest possible time-slot. The algorithm progresses from leaf to sink. Every node is scheduled only after its children node are scheduled. Thus node could perform aggregation and forwarding in the same TDMA cycle.

The algorithms presented in [7] and [8] are variations of DICA[6]. In [7] use of multiple paths between sensors and sinks is suggested. Whereas in [8] it is suggested that in addition to multiple paths, multiple channels should be used. Usage of multiple paths provides fault tolerance and that of multiple channels result in smaller schedule length.

In [9], the work done in [6] is extended for heterogeneous networks. In Figure 1 the core idea behind Attribute Aware parent selection proposed in [9].

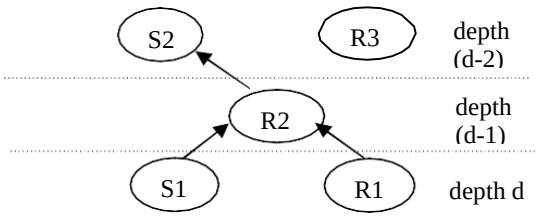

Figure 1: Illustration of Attribute Aware Parent Selection

There two types of sensors S and R shown in Figure 1. S1 and R1 both have selected R2 as parent. Packet generated by R1 will be aggregated at R2. But packet generated by S1 can not be aggregated at R2. So, R2 will send out two packets. One of type R and the other of type S. If it selects S2 as parent, S2 can not aggregate packet of type R. So, S2 will send out two packets one of type S and the other of type R. Also R3 is generating its own packet. Thus two packets coming out of S2 and one coming out of R3. Total 3 packets coming out from depth (d-2).

If node R2 selects two different parents, situation is different. It could select S2 as parent to forward packets of type S. The node R3 would be selected to forward packets of type R. Thus from depth (d-2) total two packets would come out. The node S2 would aggregate packet of type S with its own packet. Node R3 would aggregate packets of type R with its own packet.

From above explanation, it can be concluded that attribute aware parent selection would reduce the total number of packets flowing in the network. As a result, other parameters like total slots used to schedule the network, schedule length and energy consumption would also be reduced.

III. Attribute Aware Parent Selecttion

As we have used Attribute Aware Parent Selection [9] in this paper, the steps of the same algorithm are described below.

Let us assume that given node is at '*d*' hop distance from sink node. That is, its depth is *d*. It executes following steps to select one or more slots.

1. Wait for neighbor nodes at depth (d+1) to get scheduled. Once all neighbors at depth (d+1) are scheduled, go to step 2.
2. Find the number of outgoing packets and type of each packet based on type of incoming packets.
3. For each outgoing packet of type 't', follow the steps given below to select a suitable slot and parent.
4. Select the lowest slot T which is higher than time slots used by children and no neighbor is receiving or overhearing in that slot.
5. Create a candidate parent set. It is the set of neighbor nodes which do not transmit, receive or overhear in slot T.
6. If candidate parent set is empty, repeat from step 4 for next higher slot (i.e. T+1). Otherwise go to next step.
7. If any neighbor of node n in candidate parent set is of type t, it is selected as parent. If multiple such

nodes are present, select the node with lowest ID. If no such node is present, step 8 is followed. Else go to step 11.

8. Find the neighbor of given node which receives maximum number of packets of type 't'. Select that neighbor as parent. If multiple such nodes are present, select the node with lowest ID. If no neighbor of given node is scheduled to receive packets of type 't', got to step 9. Else go to step 11.

9. Find the neighbor of given node which has maximum number of neighbors of type 't'. Select that neighbor as parent. If multiple such nodes are present, select the node with lowest ID. If no neighbor of node n has any neighbor node of type 't', step 10 is executed. Else go to step 11.

10. Select the node with minimum number of unscheduled nodes as parent.

11. Broadcast REQUEST packet. It contains selected parent's ID, selected slot number and type of packet to be sent in that slot.

12. If positive RESPONSE comes from all the candidate parents, broadcast CONFIRM message to confirm the slot and parent selection.

13. If any one candidate parent sends negative RESPONSE, repeat all the steps from step 5 for next higher slot (i.e. T+1).

14. Upon reception of CONFIRM message, all the candidate parents broadcast FORBIDDEN message.

The reader is encouraged to refer [6] and [9] to find the minor details of scheduling and parent selection in heterogeneous networks. The steps 7 to 9 represent attribute-aware parent selection. They are likely to result in better aggregation.

The algorithm proposed in [9] is named here as AAJST (Attributed Aware Joint Scheduling & Tree formation). It extends DICA[6] by modifying parentselection mechanism. Its performance is compared with DICA modified for selection of multiple slot and parent pairs. The modified form of DICA is termed as DICA_Extesion. The DICA is originally proposed for selection of single pair of slot and parent. But, DICA_Extension selects multiple pairs of slot and parent. But the parent selection does not consider packet type like AA_DICA. The parent selection criteria is same as DICA. That is, the candidate parent with minimum number of unscheduled neighbors is selected as parent.

As mentioned in Section I, here both AAJST and DICA_Extension are evaluated using simulations on random node deployment.

IV. WORK DONE

A. *Simulation Design*

As mentioned earlier, algorithm presented in [9] is evaluated using random node deployment in this work. The evaluation is based on simulations. We have used Network Simulator 2 (NS 2.35) as simulation tool. The Table 1 summarizes the simulation parameters.

Table I. Simulation Setup

Parameter	Value
Area	3000m x 3000m
Inter-nodal distance	15 m
Transmission Power Consumption	0.660W
Receive Power Consumption	0.395W
Sleep Power Consumption	0W
Data Generation Rate	1 packet every 10 seconds
Simulation time	2500 Seconds

The simulation area is of 3000m x 3000m. It is divided into grid of 20m x 20m. Two horizontal or vertical grid points are 15m away. At every grid point, node is present with probability 0.5. Thus nodes are randomly deployed in the square region. Total simulation time is 2500 seconds. In the first 2000 seconds, scheduling and tree formation algorithm takes place. In the last 500 seconds, data transmission takes place.

The performance of the algorithm is measured with respect to increase in No. of Attributes. When No. of Attributes is 1, network is considered as homogeneous network. As number of attributes increases, network becomes more and more heterogeneous. Here 'Attribute' means 'type'. Thus when number of attributes is 2, it means there are two different types of nodes are present in the network. If there are A_n attributes present in a network, a node is assigned a attribute with probability $1/ A_n$. That is, if there are 4 types of nodes are present, probability that a node is of particular type is 1/4 i.e. 0.25.

Following performance parameters are considered:

a. Schedule Length: It is the count of unique time-slots used to schedule the entire network.
b. Average Aggregation Factor: Aggregation factor at a node is ratio of count of received minus forwarded packets and count of received packets by that node. Average Aggregation Factor is an average of Aggregation Factors taken over all the nodes.
c. Control Overhead: It is the count of REQUEST,

REPLY, FORBIDDEN and CONFIRM messages exchanged among the nodes.

d. Energy Consumption during Control Interval: It is the amount of energy consumed due to control messages transmission. First, energy consumption at each node during control interval is found. Then an average is taken over all the nodes.

e. Energy Consumption during Data Interval: It is the amount of energy consumed due to transmission of data packets. Like energy consumption during control phase, this is also an average taken over all the nodes.

B. Results & Discussion:

In Figure 2, graphs of Average Aggregation Factor v/s. Number of Attributes are presented. It is seen from the graphs that as number of attributes increases, aggregation factor is reduced. At heterogeneity increases, aggregation becomes poor. So, more packets are forwarded. Thus aggregation factor goes down. As the proposed method does parent selection such that packet gets aggregation as early as possible in its path towards sink, the proposed method results in better aggregation factor than the DICA_Extension.

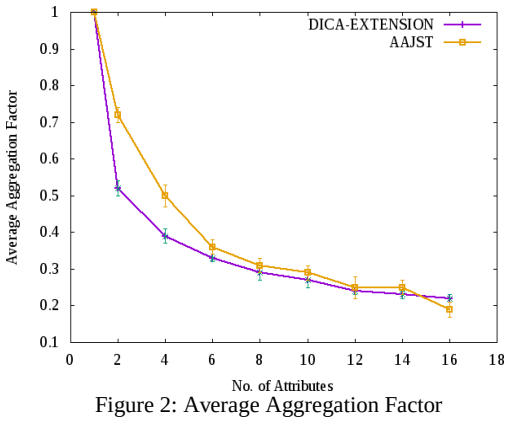

Figure 2: Average Aggregation Factor

The AAJST results in 20% better Average ggregation Factor than DICA_Extension when Number of Attributes is 2. Gradually the difference between the two algorithms decreases. When number of attributes is 16, the AAJST results in 3% better aggregation factor than DICA_Extension.

In Figure 3, graphs of Total number slots v/s. Number of Attributes are presented. As heterogeneity increases, aggregation becomes poor. So, more packets are forwarded by nodes. As a result, more slots are required to schedule the network. As the proposed algorithm results in better aggregation compared to DICA_Extension, it results in lesser transmission slots than DICA_Extension. It can be observed that the AAJST algorithm results in around 10% reduction in total count of transmission slots.

The schedule length is the count of unique slots used to schedule the network. As total transmission slots increases, schedule length is also increases. The same thing is reflected in Figure 4.

The AAJST and DICA_Extension result in almost the same schedule length when number of attributes is 2,4 and 6. As number of attributes increases, the difference

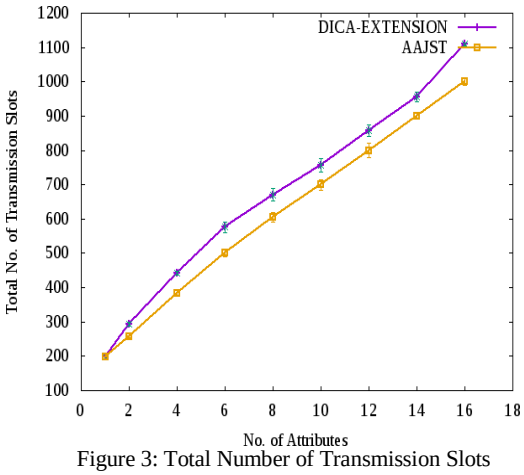

Figure 3: Total Number of Transmission Slots

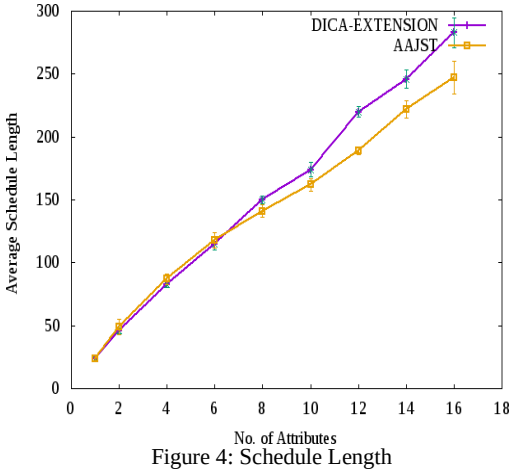

Figure 4: Schedule Length

also increases upto 10%.

The graphs of Control Overhead v/s. Number of Attributed are presented in the Figure 5. The control overhead is proportional to number of slots required to schedule the network. As number of transmission slots increases with number of attributes, the control overhead also increases with number of attributes. As the proposed algorithm results in lesser transmission slots than the DICA_Extension, it also results in smaller control overhead. The control overhead is reduced by 7% to 10% in AAJST as compared to DICA_Extension.

The energy consumption during control interval is proportional to control overhead. It is the energy spent

in sending and receiving control messages. As control overhead increases, energy consumption also increases.

The graphs for energy consumption during control phase are shown in the Figure 6. The nature of the graphs is same as those in Figure 5. The energy consumption during control phase is reduced by 5% in AAJST as compared to DICA_Extension.

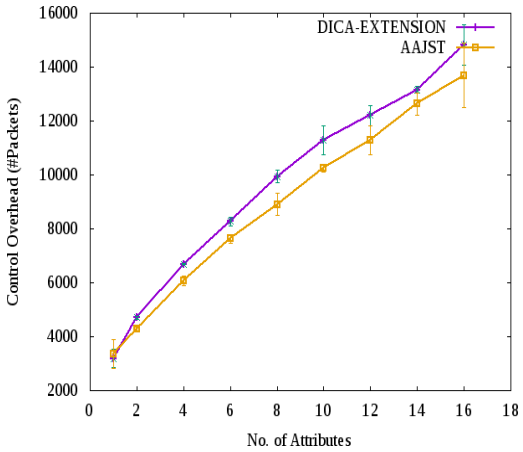

Figure 5: Control Overhead

In the Figure 7, graphs of Energy Consumption during Data Phase v/s. Number of Attributes are shown. As the proposed algorithm results in better aggregation, number of packets forwarded per node is also reduced. So, energy spent in sending (and also receiving) packets is reduced. Thus the proposed algorithm results in lesser energy consumption during data phase. But as aggregation deteriorates with increase in heterogeneity, energy consumption during data phase also increases with increase in number of attributes.

The AAJST algorithm result in reduction in energy consumption between 15% to 30%.

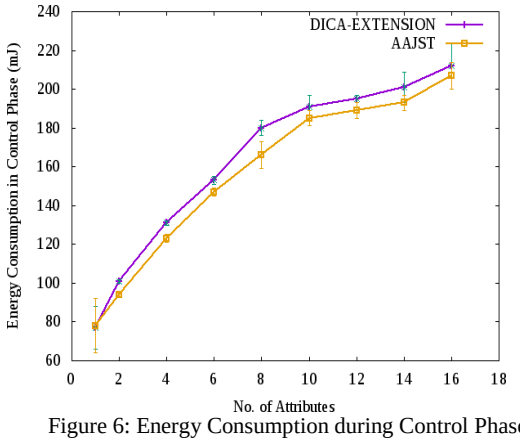

Figure 6: Energy Consumption during Control Phase

V. Conclusion

The AAJST algorithm results in improvement in aggregation. As a result, the number of packets passing through the network is reduced. This results in reduction in the total count of slots required to schedule the network and schedule length. Also control overhead and corresponding energy consumption is reduced. As number of packets sent/received by node is reduced, data energy consumption is also reduced.

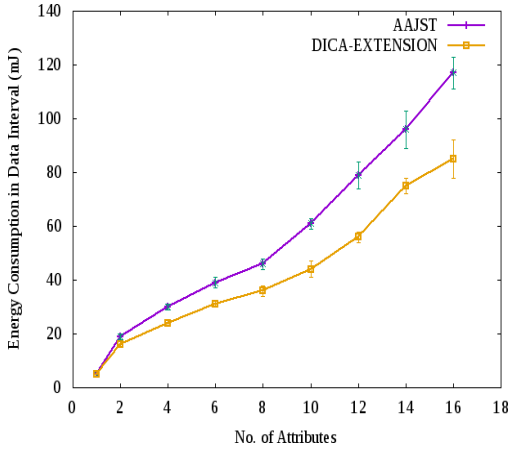

Figure 7: Energy Consumption during Data Phase

The reduction in schedule length is going to result in the time for which node has to wait for its transmission turn. Suppose, schedule length of a tree is T. So, every node will get its transmission turn after (T-1) slots. If schedule length is small, node will get transmission turn more frequently and the packets would not be buffered for longer. As a result, packet latency would be reduced. As AAJST results in smaller schedule length, it can be said that AAJST would also result in reduction in packet latency.

It is seen from the results the AAJST algorithm results in reduction in energy consumption during control phase and data phase. Sensor nodes always have limited energy. The reduction in energy consumption results in extended lifetime of nodes. As a result, network is not partitioned and remains connected. Thus AAJST is likely to result in better network lifetime.

It can be concluded that AAJST algorithm improves packet delivery latency and network lifetime.

References

[1] F.Ren et. Al, “Attribute-aware Data Aggregation Using Potential Based Dynamic Routing in Wireless Sensor Networks”, in IEEE Transactions on Parallel and Distributed Computing, Vol. 24, Issue 5, 2012.

[2] Amitabha Ghosh et. al, “Scheduling Algorithms for Tree-based Data Collection in Wireless Sensor Networks”, in *Theoretical Aspects of Distributed Computing in Sensor Networks*, pp. 407-445,

Springer, 2010.

[3] Ichrak Amdouni et. al, "Joint Routing and STDMA-based Scheduling to Minimize Delays in Grid Wireless Sensor Networks", A Research Report, September 2014.

[4] Fang-Jing Wu et. al, "Distributed Wake Up Scheduling for Data Collection in Tree based Wireless Sensor Networks", in *IEEE Communication Letters*, Vol. 13, Issue 3,2009.

[5] Chansu Yu et. al, "Many to One Communication Protocol for Wireless Sensor Networks", in *International Journal of Sensor Networks*, Vol. 10, Issue 10, 2012.

[6] M.Bagga et. al, "Distributed Low Latency Data Aggregation Scheduling in Wireless Sensor Networks", in *ACM Transactions on Sensor Networks*, Vol. 11, Issue 3, April 2015.

[7] M.Bagga et. al, "Efficient Multi-path Data Aggregation Scheduling in Wireless Sensor Networks", in *proceedings of IEEE International Conference on Communications*, 2013.

[8] M.Bagga et. al, "Multi-path Multi-Channel Data Aggregation Scheduling in Wireless Sensor Networks", in *proceedings of IEEE Wireless Days International Conference*, 2013.

[9] T. Vasavada et. al, "Joint Distributed Scheduling and Tree formation for Heterogeneous Wireless Sensor Networks", in *proceedings of IEEE Advance Networks and Telecommunication Systems (ANTS)*, November, 2016.

AUTHOR PROFILE

Dr. Tejas Vasavada

Dr. Tejas Vasavada is an Assistant Professor at Lukhdhirji Engineering College (managed by Govt. of Gujarat), Morbi, Gujarat, India. His areas of interests include ad hoc & sensor networks, algorithm design and simulation & modeling.

Dr. Sanjay Srivastava

Dr. Sanjay Srivastava is a Professor at Dhirubhai Ambani Institute of Information and Communication Technology (DA-IICT), Gandhinagar, India. He has received his PhD from University of California, Los Angeles. He works in the area of sensor networks and network protocol design & analysis.