IRACST – International Journal of Computer Networks and Wireless Communications (IJCNWC), ISSN: 2250-3501
Vol.4, No5, October 2014

An Energy-Efficient Data Aggregation Tree Construction Algorithm for Wireless Sensor Networks

Md. Sajidul Islam, Imtiaz Bin Rahim and Mosarrat Jahan
Department of Computer Science and Engineering
University of Dhaka,
Dhaka, Bangladesh
Email: sajidulislam52@yahoo.com, evancsedu09@yahoo.com, mosarrat@cse.univdhaka.edu

***Abstract—* In this paper, we study the problem of constructing an energy-efficient data aggregation tree for wireless sensor networks. Our aim is to reduce the cost of transmitting data by aggregating data at appropriate intermediate nodes. We proposed a polynomial time algorithm Energy-Efficient Data Aggregation Tree (EEDAT), to reduce the transmission cost by decreasing the hop distance a packet will travel and by merging data into suitable intermediate nodes. For sending data from a node at level l, EEDAT algorithm intelligently selects a node at level l-1 by using the aggregation ratio associated with every sensor node. We have conducted extensive simulations for analyzing the performance of EEDAT and compare the result with existing Tree-Adapting [1] algorithm. Our experiments show that EEDAT exhibits better performance as compared to Tree-Adapting algorithm in reducing the total energy cost due to packet transmissions.**

Keywords-wireless sensor networks; data aggregation tree; aggregation ratio;

I. INTRODUCTION

Wireless Sensor Networks (WSNs) are widely exploited for pervasive computations that often involve observing vast space through the usage of a large number of sensor nodes. Lifetime of WSNs is restricted by the energy-constrained nature of sensor nodes. A sensor node is equipped with a limited energy battery due to its size constraint and replacing a sensor node is not possible when its energy depleted as it is often exploited in a difficult to access regions [2]. To address this issue, high volumes of research works are conducted on designing energy-efficient algorithms with the aim of saving energy. One way to conserve energy is to aggregate data from various nodes to a sink node at intermediate nodes. Data coming from a number of nodes are combined at intermediate nodes in their route to the sink node and often data are merged or filtered through a number of operations in order to avoid data redundancy [1]. Significant energy savings can be achieved by carefully selecting intermediate nodes to merge data. In this paper we address the problem of constructing data aggregation tree to reduce the communication cost by either reducing the number of data transfers or by reducing the hop count data will travel. In the next section we formalize the problem.

A. Problem Formulation

A *Wireless Sensor Network* is represented as a graph $G = (V, E)$ where V is the set of sensor nodes and $E \subseteq V \times V$ is the set of links. We consider a WSN G where each node $v \in V$ is associated with a weight $w(v) \in Z_0^+$, that represents the size of data to be reported by v to a sink node s. Our goal is to construct a *Data Aggregation Tree* that will minimize the communication cost and thus save energy. A data aggregation tree T is a spanning tree rooted at s and represented as $T(V_T, E_T)$ where $V_T = V$ and $E_T \subseteq E$. Every node $v \in V_T$ at level l can transmit data to its parent node u at level $l-1$. Each node $v \in V$ is accompanied with an aggregation ratio $\alpha \in Z^+$, that denotes the maximum amount of data that can be combined into one packet. Every node $v \in V_T$ can combine its own data and the data received from its child nodes in T into a number of packets depending on α and send them to its parent node u in T. Not every nodes $v \in V$ will generate data and they will act as relay nodes by passing data to other nodes. We assume that G will contain a percentage of relay nodes and it is denoted by γ. We define the cost of T denoted by $Cost(T) = \sum_{u=1}^{N}(\text{No of packets} \times \text{hop counts of the path from u to s})$. So the problem we are interested in is the *Energy-Efficient Data Aggregation Tree (EEDAT)* problem: given a WSN $G = (V, E)$ with $w(v) \in Z_0^+$ associated with each node $v \in V$ and a sink $s \in V$, an aggregation ratio $\alpha \in Z^+$ and percentage of relay nodes γ, find a data aggregation tree T rooted at s such that $Cost(T) \leq Cost(T')$ for all data aggregation tree T' of G.

B. Our Contribution

In this paper, we examine the problem of finding energy-efficient data aggregation tree for WSNs. We propose a polynomial time algorithm: Energy-Efficient Data Aggregation Tree (EEDAT) Algorithm. Our experimental results exhibit that EEDAT is capable of reducing the energy cost in every case as compared to the Tree_Adapting Algorithm [1].

The rest of this paper is organized as follows. In Section II we elaborate the related works and in Section III we present the EEDAT algorithm in detail. We explain the simulation results in Section IV. Finally we conclude the paper in section V with some future directions for research.

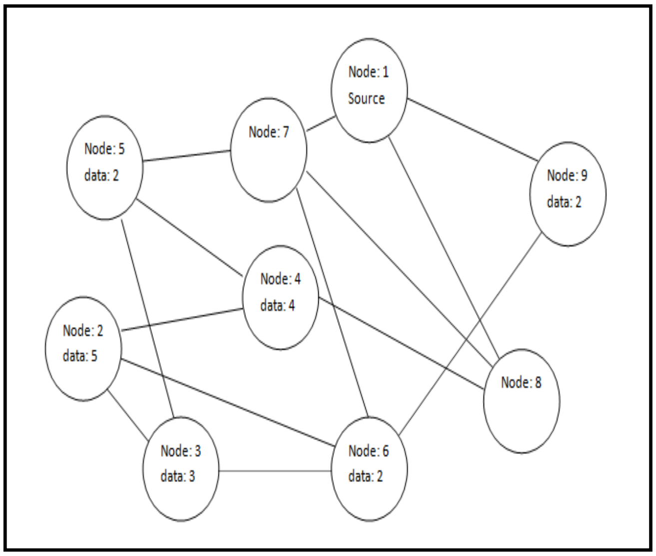


Figure 1. Wireless Sensor Network G with 9 Nodes

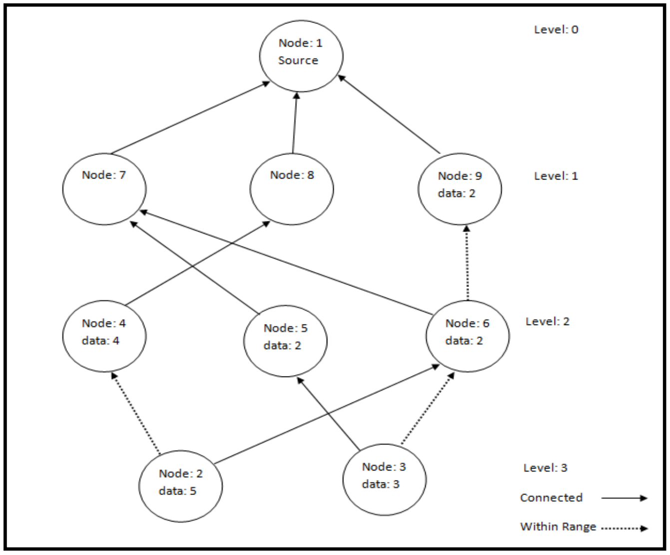


Figure 2. Leveled Tree Generate by EEDAT

II. RELATED WORK

Data aggregation is an efficient mechanism to save energy in resource-constrained WSNs and is widely studied in literature. Zhang et el. [1] proposed Tree-Adapting algorithm to minimize energy expenditure due to data communications. This algorithm reorganizes an existing data gathering tree in order to reduce communication cost. The authors used two techniques: (1) Tree Tagging and (2) Father Adjustment and Operator Placement. In tree tagging, the base station marks itself with a tag value v of 0 and broadcasts a message with this tag value to its 1-hop neighbors in the gathering tree. The 1-hop neighbors of the transmitting node will mark themselves as $v = v + 1$ if they are not generating data. Otherwise mark themselves as 0. This procedure continues until all nodes of the network received a tag value. After tagging the tree, a node selects a new parent if the tag value of the new node is smaller among all the 1-hop neighbors that are not its children in the gathering tree and less than the tag value of the current parent. If a node finds more than one flow passing through it, it will aggregate data. We compare the performance of our proposed algorithm with Tree-Adapting [1] algorithm. The authors of [3] proposed a reconstruction method to recreate a similar tree from the old one that has lower energy consumption. The algorithm starts examining the nodes u from the bottom level of tree T and checks whether an edge (u, v) not in T is advantageous in terms of energy reduction in place of another edge (u, w) in T and if so, (u, v) is included in T and (u, w) is removed. The authors applied the reconstruction mechanism on Shortest-Path-Tree and named the new tree as Reduced-Shortest-Path-Tree. The experimental results show that Reduced-Shortest-Path-Tree performs better in terms of total energy reduction. Weng et el. [4] proposed algorithms to solve data aggregation problem using multiple spanning trees. The experimental results showed that multiple spanning trees can save more energy as compared to single spanning tree for data aggregation. The authors transformed the problem of constructing multiple spanning trees into a linear programming problem of data flow network and derived an optimal rate between two adjacent sensor nodes. The spanning trees are constructed using Breadth-First Search and Depth-First Search algorithm, respectively. In [5], the authors proposed an algorithm LPT that choose a node with highest energy as root node. A node always selects a node as parent that has maximum energy and in case of tie, the node closest to the root in terms of distance. LPT also considered a third parameter that is the number of children of a node in the tree so that each node in the tree can have equal number of children. The authors of [6] proposed a secure and energy-efficient data aggregation protocol known as Energy-Efficient Secure Pattern based Data Aggregation (ESPDA). This protocol protects redundant data transmission from sensor nodes to cluster-heads by using pattern code. If a number of sensor nodes sense the same data, only one of them is kept in active state while other nodes are kept in sleep mode. The active node generates pattern codes to represent the characteristics of sensed data and sends it to cluster-head that performs data aggregation based on pattern codes. Thus only distinct data in encrypted form is sent from sensor nodes to the base station. As the cluster head does not know actual sensor data, security of data is ensured. Performance analysis shows that ESPDA achieves 50% bandwidth efficiency as compared to the conventional algorithms.

III. ALGORITHM

In this section we present our proposed polynomial time heuristic algorithm.

A. *Energy Efficient Data Aggregation Tree (EEDAT)*

At first this algorithm labels the network using Breadth-First Search (BFS) algorithm [7] starting from root and accordingly a BFS tree is generated. A node with data at level

Algorithm 1 EEDAT Algorithm for Data Aggregation Tree Construction

```
Input: A WSN $G=(V, E)$ and $s \in V$

Output: Aggregation Tree $T$

procedure GENERATE_LEVEL($G$, $s$)
    for all $u \in V$ do
        $u.level := \infty$
        $u.color :=$ WHITE
    end for
    $Q := \varphi$
    $T := \varphi$
    $Height := 0$
    $s.level := 1$
    $s.color :=$ GREY
    ENQUEUE($Q$, $s$)
    $T := T \cup \{s\}$
    while $Q \neq \varphi$ do
        $u :=$ DEQUEUE($Q$)
        for all $v \in Adj[u]$ do
            if $v.color ==$ WHITE then
                $v.level := u.level + 1$
                $v.color :=$ GREY
                $T := T \cup \{v\}$
                $Height := v.level$
                ENQUEUE($Q$, $v$)
            end if
        end for
        $u.color :=$ BLACK
    end while
end procedure

procedure SEND_DATA($u$, $Cost$, $\alpha$)
    $Neighbor(u) := \{v \mid v \in l - 1$ and $(v, u) \in E\}$
    $Child(v) := \{u \mid u \in l$ and $(v, u) \in E\}$
    for all $v \in Neighbor(u)$ do
        for all $x \in Child(v)$ do
            if $data[x] \neq 0$ then
                $Priority[v]$:=Number of free slots for data
                Insert $Priority[v]$ into $Lp$ in descending order
            end if
        end for
    end for
    while $Lp := \varphi$ do
        $Priority[v] :=$ Extract($Lp$)
        if $data[u] \neq 0$ then
            Update $data[u]$
            Update $Priority[v]$
            $Cost := Cost + 1$
        end if
    end while
end procedure

procedure CALCULATE_COST($T$, $Height$, $\alpha$)
    $l := Height$
    $Cost := 0$
    while $l \neq 1$ do
        for all $u \in l$ do
            if $data[u] \geq \alpha$ then
                $Neighbor(u) := \{v \mid v \in l - 1$ and $(v, u) \in E\}$
                $Data\ Send := data[u] - (data[u] \bmod \alpha)$
                $Cost := Cost + (data[u] / \alpha)$
                $data[u] := (data[u] \bmod \alpha)$
            end if
        end for
        for all $u \in l$ do
            if $data[u] \neq 0$ then
                SEND_DATA ($u$, $Cost$)
            end if
        end for
        $l := l + 1$
    end while
end procedure
```

Algorithm 1. Pseudo code of EEDAT for Aggregation Tree Construction

l will simply sends the packets to any of its 1-hop neighbors at $l - 1$ level and tries to aggregate with any nodes at level l through level l-1. At this point, EEDAT tries to increase the rate of aggregation by choosing the correct parent with a modified node list which is a priority queue based on remaining capacity of every node at level $l - 1$. The whole network is divided into parts and calculation is always performed between l and $l - 1$ level until the root is reached. Any node at l level will check its 1-hop neighbors at $l – 1$ level and will send data into a number of passes. In each pass, a node can send maximum α numbers of data. For remaining data, a check is performed to determine best candidates among the $l - 1$neighbors. Each of $l – 1$ neighbors are examined to determine whether the node has child with remaining data at l level and the number empty slots for that particular node to merge those data and accordingly a priority is assigned to the nodes at $l - 1$ level. A node with highest priority is selected for merging data. If the default parent has same attribute as other available nodes, default parent will be selected. The pseudo code of our algorithm is shown as Algorithm 1. The complexity of *GENERATE_LEVEL(G)* is $O(N + M)$ and the complexity of *CALCULATE_COST(T, Height, α)* is $O(H(N^3 + N^2 Log(N)))$.

Fig. 1 shows a WSN G consisting of 9 nodes, where 1 is a source node. Node $2, 3, 4, 5, 6$ and 9 has $5, 3, 4, 2, 2, 2$ data, respectively to be sent to the base station 1. Node 7 and 8act as relay nodes. For this example aggregation ratio α is assumed to be 4. The working principle of EEDAT is illustrated with respect to G. Fig. 2 displays the tree leveled using BFS algorithm. Fig. 3 shows the flow of data from level 3 to level 2. At level 3, nodes 2 and 3 have 5 and 3 data to send, respectively. Node 2 can choose between two available paths: $2 \rightarrow 4$ and $2 \rightarrow 6$ and send first 4 data merged into a single packet to 6 as 6 is the parent of 2. For node 3 the available

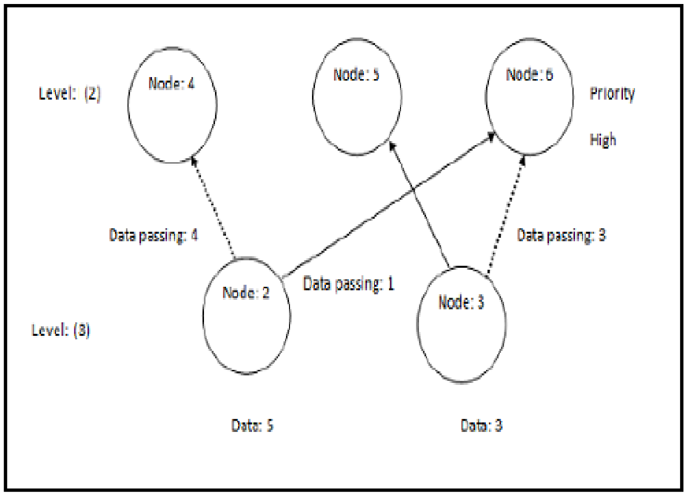


Figure 3. Data Passing from Level 3 to 2

paths are *3* → *5* and *3* → *6*. Node *3* finds that at level 2, node *6* has a child node *2* that has *1* data yet to be sent. So *3* send all *3* data to *6* to get aggregated with the remaining data of *2*. Node *2* sends it remaining *1* data to *6* and thus total cost for passing data from level 3 to 2 is *3*. Data flow from level 2 to 1 is shown in Fig. 4. At level 2 there are three nodes *4*, *5* and *6* with *4*, *2* and *10* (*8* data from level 3 and *2* data of its own) data, respectively. As node *4* has only one available path *4* → *8*, it sends *4* data to *8* and the cost will be 1. Once again only available path for *5* is *5* → *7* and *5* sends *2* data to *7* producing a cost of 1. As *2* slots are still empty at *7* the priority of *7* is high and *6* sends *2* data to *7*. Node *6* has still *8* data remaining to send with 2 available paths *6* → *7* and *6* → *9*. As there is no high priority node at level 1, *6* selects its default parent *7* to send remaining data with a cost of *2*. Thus cost of passing data from level 2 to 1 is *5*. Flow of data from level 1 to 0 is shown in Fig. 5. Level 1 contains 3 nodes 7, 8 and 9 with 12, 4 and 2 data, respectively that are to be forwarded to the base station. 12 data from node 7 are sent using 3 packets. Similarly, node 8 sends its 4 data with a cost of 1 and 9 sends 2 data with a cost of 1. So the total cost is 5 for sending data from level 1 to level 0. Thus the total cost of sending data from all nodes to the base station 1 is 3+5+5=13.

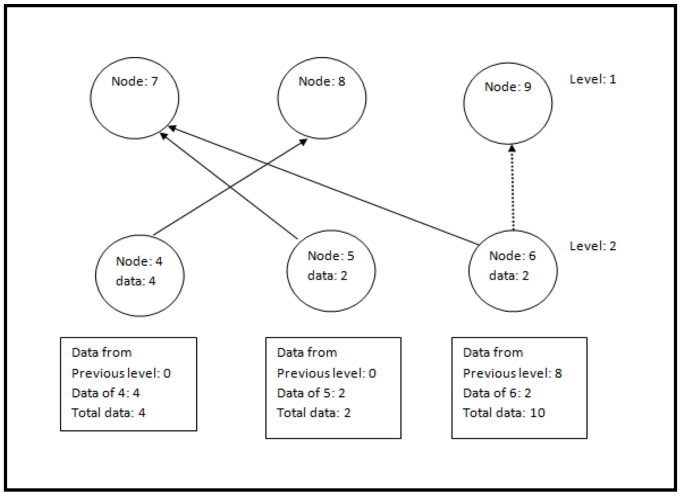


Figure 4. Data Passing from Level 2 to 1

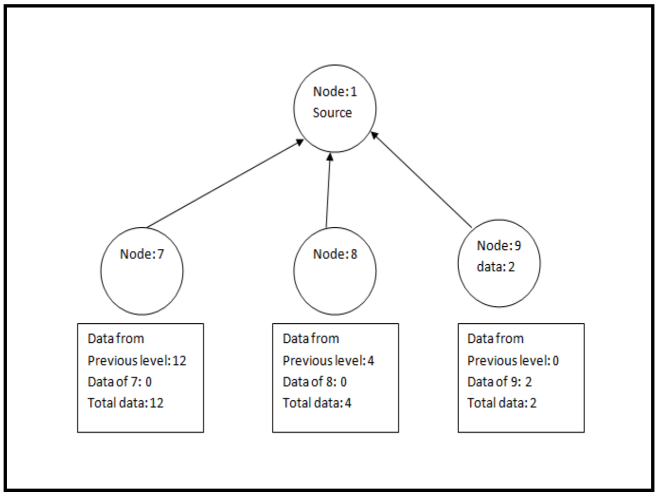


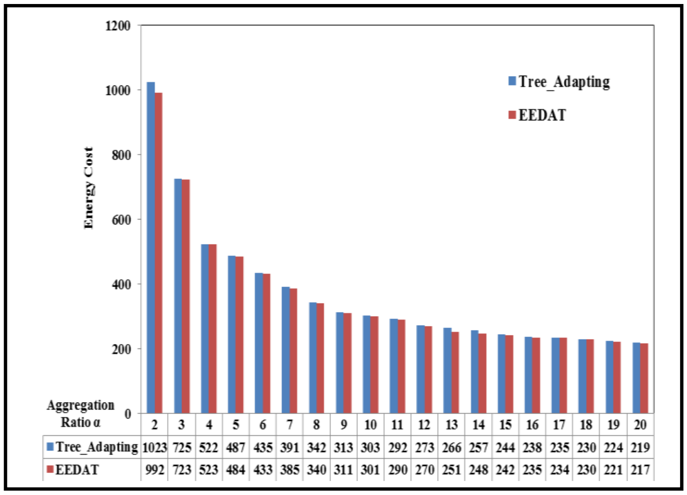
Figure 5. Data Passing from Level 1 to 0

IV. SIMULATION RESULTS

We conducted extensive simulation in order to study the performance of EEDAT algorithm. We compared our results with existing Tree-Adapting algorithm [1]. The performance metric used was the total energy cost calculated as

$$\sum_{u=1}^{N}(\text{No of packets} \times \text{hop counts of the path from } u \text{ to } s)$$

We used C++ platform to simulate our proposed algorithm. The simulations were conducted on 30 connected networks generated randomly; each consists of 400 nodes deployed in an area of 200m × 200m. An edge exists between node *u* and *v* if and only if the Euclidean distance between them is less or equal to the transmission range. We assume on average every node can have 8 neighbors.



Aggregation Ratio α	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
🟦 Tree_Adapting	1023	725	522	487	435	391	342	313	303	292	273	266	257	244	238	235	230	224	219
🟥 EEDAT	992	723	523	484	433	385	340	311	301	290	270	251	248	242	235	234	230	221	217

Figure 6. Energy Cost for EEDAT and Tree_Adapting Algorithm for various values of α when γ = 50% and β = 3

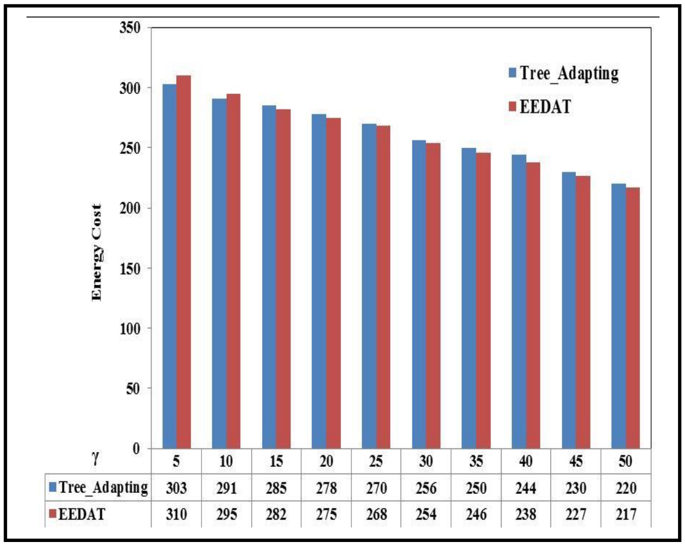


Figure 7. Energy Cost for EEDAT and Tree_Adapting Algorithm for various values of γ when α = 20 and β = 3

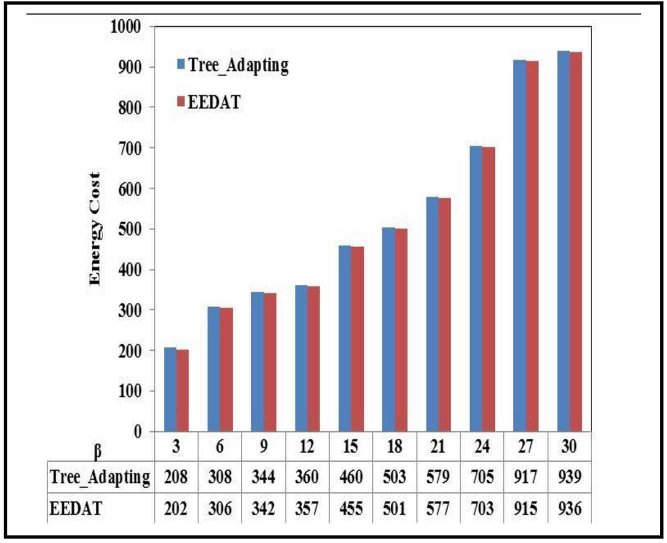


Figure 8. Energy Cost for EEDAT and Tree_Adapting Algorithm for various values of β when α = 20 and γ = 50%

Transmission range of node *u* was calculated as $\sqrt{(8 \times 200 \times 200)/(400 \times \pi)}$. Every node in the network was assigned an integer weight from the interval [1, β] if the node was not a relay node. Otherwise, the node was assigned a weight 0. Here β denotes maximum data size and we altered the value of β from 3 to 30. Relay nodes do not generate data but can merge received data and can transmit them. In our experiment, γ denotes the percentage of nodes act as relay nodes. Values of γ were selected in the range 5 to 50. Every node in the network is associated with an aggregation ratio denoted by α. We altered the value of α between 2 to 20.

Fig. 6 depicts the total energy consumed by EEDAT and Tree_Adapting algorithm under different values of α, where γ = 50% and β = 3. As α increase, energy cost gets reduced for both algorithms. Larger values of α creates the provision of accommodating more data into a single packet and as the number of packets decreases, the energy cost reduces ultimately. Although the performance of both algorithms are similar, EEDAT generates least cost in all of the cases, especially for α = 2, EEDAT generates energy cost of 992 whereas Tree_Adapting algorithm produces a cost of 1023. Similarly for α = 13, the cost generated by Tree_Adapting algorithm and EEDAT is 266 and 251, respectively.

Fig. 7 shows the performance of EEDAT and Tree_Adapting algorithm under different values of the γ, where α = 20 and β = 3. Performance improves for both algorithms with increasing number of relay nodes. This is natural because more amounts of relay nodes will decrease the number of nodes generating data. Although for smaller values of γ the performance of EEDAT is not better than Tree_Adapting algorithm, with increasing values of γ, EEDAT performs better. For γ = 40, the cost generated by EEDAT and Tree_Adapting are 238 and 244, respectively.

Fig. 8 exhibits the comparison of the total energy costs between the EEDAT and the Tree_Adapting algorithm for different values of β where γ = 50% and α = 20. EEDAT generates reduced cost as compared to Tree_Adapting. For β = 30, EEDAT produces a cost of 936 whereas Tree_Adapting generates 939.

V. CONCLUSION

In this paper, we study the problem of constructing energy-efficient data aggregation tree for WSNs in order to reduce the energy required for data transmissions. We developed a polynomial time algorithm that wisely selects intermediate nodes based on remaining capacity of a node to fill up a data packet and merges data into a moderate number of packets. Our algorithm shows better result than Tree_Adapting algorithm for increasing values of β and γ and exhibits almost similar performance asTree_Adapting algorithm for α. We are interested to refine the selection criteria of nodes to send data in a more intelligent way so that we can aggregate more data using less number of data packets. Our algorithm is centralized and in future we are interested to implement the distributed version and analyze its performance.

REFERENCES

[1] J. Zhang, H. Peng, and T. Yin, "Tree-adapting: an adaptive data aggregation method for wireless sensor networks", in *Proc. 6th Int. Conf. Wireless Communications Networking and Mobile Computing (WiCOM'10)*, pages 1-5, Chengdu,China, September 2010.

[2] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "A survey on sensor networks", *IEEE Communications Magazine*, Vol. 40, No. 8, pages 102–114, August, 2002.

[3] B. H. Liu, Y. H. Jhuang, L. P. Tung, and J. Y. Jhang, "An improved method of constructing a data aggregation tree in wireless sensor networks", in *Proc. 6th Int. Conf. Genetic and Evolutionary Computing,* pages 344–347, Kitakushu, Japan, August 2012.

[4] C. Weng, M. Li, and X. Lu, " Data aggregation with multiple spanning trees in wireless sensor networks", in *Proc. Int. Conf. Embedded Software and Systems (ICESS2008)*, pages 355–362, Sichuan, China, July 2008.

[5] Z. Eskandari, M. H. Yaghmaee, and A. H. Mohajerzadeh, "Energy efficient spanning tree for data aggregation in wieless sensor networks", in *Proc. 17th Int. Conf. Computer Communications and Networks (ICCCN2008)*, pages 1–5, St. Thomas, US Virgin Islands, USA, August 2008.

[6] H. Cam, S. Ozademir, P. Nair, D. Muthuavinashiappan and H. O. Sanil, "Energy-efficient secure pattern based data aggregation for wireless sensor networks", *Elsevier Computer Communications*, Vol. 29, No. 4, pages 446-456, February 2006.

[7] E. F. Moore, "The shortest path through a maze", in *Proc. Intl. Symposium on the Theory of Switching*, pages 285–292, Harvard University Press, 1959.

AUTHORS PROFILE

Md. Sajidul Islam received BSc (Hons.) Degree in Computer Science and Engineering from University of Dhaka, Bangladesh in 2014. He is currently pursuing MSc. degree in Computer Science and Engineering in University of Dhaka. His field of interest includes wireless sensor network, image processing, Artificial Intelligence, software Architecture, Web Applications

Imtiaz Rahim received B.Sc (Hons.) in Computer Science and Engineering from University of Dhaka, Bangladesh in 2014. He is currently continuing his M.Sc degree in Computer Science and Engineering in University of Dhaka. His research interest includes computer network, wireless sensor network, data mining and game development.

Mosarrat Jahan received B.Sc (Hons.) and M.S degree from University of Dhaka, Bangladesh and M. Comp.Sc degree in Computer Science from Concordia University, Canada in 2012. Currently she is working as an assistant professor in Department of Computer Science and Engineering, University of Dhaka. Her research interest includes energy efficient data communications in wireless sensor networks, content management in mobile ad hoc networks and network security.