© 2019 IJSRST | Volume 6 | Issue 4 | Print ISSN: 2395-6011 | Online ISSN: 2395-602X
Themed Section: Science and Technology
DOI : https://doi.org/10.32628/IJSRST196462

Improving Wireless Sensor Network Lifetime Using Self-Organizing Protocol

Charan Mangali
Department of Electronics and communication Engineering, J.N.T.U College of Engineering, Anantapur, Andhra Pradesh, India

ABSTRACT

This paper proposes a Efficient Tree-based Self-organizing Protocol to improving wireless sensor network lifetime. all nodes are divided into two kinds: network nodes and non-network nodes Network nodes broadcast packets to select child nodes and non-network nodes collect packets to join in network. During the self-organization process we take hop, residual energy, number of child node and communication distance into account to calculate the weight of available sink nodes, and then select the node with max weight as sink node. After a non-network node joins the network successful it will work as a network node to searching child nodes. A tree network can be constructed one layer by one layer. For balancing energy consumption and prolonging network lifetime we adjust the topology dynamic. All experiments were done with NS2 Furthermore, the success rate of packet and Life time is much higher compared with LBT.

Keywords : Internet of Things, Self-organization, Tree-based Sensor Networks, Lifetime

I. INTRODUCTION

Fueled by the increasing capability and decreasing cost of wireless sensors, wireless sensor networks (WSNs) hold the promising applications for battlefield surveillance and environmental data collection (e.g., temperature, humidity, vibrations, seismic events, etc.) [1]. In such applications, a large number of sensors are dispersed over the monitoring area to collect the needed information. During the lifetime of the WSNs, the sensed data is periodically gathered and transmitted to the sink node for further processing [2]. On the other hand, the sensor nodes in WSNs generally rely on batteries with very limited power. Moreover, WSNs are often deployed in remote or inaccessible environments, where the recharging or replacement of batteries would be a mission impossible [3]. Therefore, to enable the networks to work longer, energy-efficient designs for data gathering are of paramount importance.

According to the applications of WSNs, the data gathering can be categorized into two types [4], namely data gathering with aggregation and without aggregation For data gathering with aggregation, the sensed data can be merged (e.g., SUM, MAX, MIN, etc.) into a much smaller size [5]. For example, to collect the highest temperature in a monitoring area,the relay nodes compare all the received temperature values and only send the maximum one to their parents Data gathering without aggregation refers to the opposite case. All the sensors simply transmit all the raw data of their own as well as their children to their parents without data compression.ss Ye et al. in [6] have verified that without data aggregation the upper limit of all one-hop nodes' energy consumption is 98%. LBT (Load-Balanced and energy-efficient Tree) can maximize the network lifetime. Authors take load-balancing and energy efficient of one-hop nodes into account to construct the tree based network. Algorithm LBT can preserve

that the energy consumption of the tree-based network is close to the upper limit, approximately. Data aggregation technology isn't used in above literature. So these methods increase the energy consumption and network load when data aggregation occurs. In this paper we use the data aggregation technology in tree based network to reduce the energy consumption and network load

In this paper, an Self-organization Protocol in tree-based network is proposed. The network nodes (the nodes that have joined the network) are classified into three types: root node, sink node, sensor node. In the beginning there is only a root node whose hop is zero. Then, the root node searches child nodes by broadcasting packets. After receiving the broadcast packets, the neighboring non-network nodes record the topology information and use different metrics such as number of child nodes, hop, communication distance and residual energy to reach available sink nodes' weight. Next, the node with max weight is selected as sink node. When non-network nodes join the network successfully, they can be turned into network nodes at once. Our proposed algorithm can build a tree-based network quickly. In addition, we adjust the topology dynamically and remove the farthest child node to balance energy consumption and prolong the whole network lifetime.

The rest of this paper is organized as follows: Section 2 analyses the algorithm Self-organization Protocol. Section 3 is the implement of Section 2. The experiments and experimental results are discussed in Section 4. Section 5 is the conclusion of this paper.

II. SELF-ORGANIZING PROTOCOL

A. Network Self-organization

Eq. 1 is an energy model [7]. Assuming that the distance between $node_i$ and $node_j$ is d and the packet length is L bits, the energy cost of sending L bits data is $E_{t,j}$ (L, d) and $E_{r,x}$(L, d) is the energy cost of receiving L bits data.

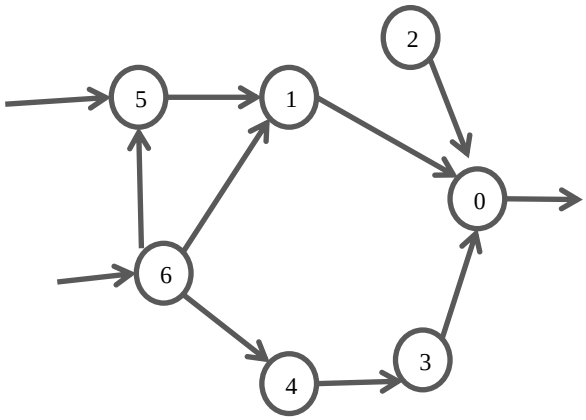

Fig.1: A sensor network topology

E_c is the basic energy consumption of send-receive link. d_{cr} is the threshold of communication distance. e_1 and e_2 are energy units, corresponding to $d < d_{cr}$ and $d > d_{cr}$. The result of this energy model is determined by d in practical application. So we can select the closest node as sink node, which is good for reducing the energy consumption.

$$E_{i,j\ t}(L, d) = E_{t,x}(L, d) + E_{r,x}(L, d) = L(2E_c + ed^k)$$

In addition, we also consider the average hop of network when selecting sink node.Above figure 1 is a sensor network topology. Node6 can communicate with Node5 and Node1. Node6 is closer to Node5. If Node6 selects Node5 as sink node its hop is 2, else if Node6 selects Node1 as sink node its hop is 1. We assume that d_1 is 6m, d_2 is 7m, d is 12m, according to Eq. 1 Node6 selects Node5 as sink node is more . The more child nodes the more received packet at the same time and it will consume more energy. So we need to balance the hop, residual energy, number of child nodes and distance of node to select the best sink node. Referring to (5), (6) of [9] and balancing these four factors we get (1) to calculate the weight of sink nodei is the weight of nodei and it is the larger the better. Di is the distance between current node

with nodei and it is the smaller the better. Ni is the number of child node of nodei and it is the smaller the better. (Ni+1) is the degree of nodei. Ei is the residual energy of nodei and it is the greater the better. Hi is the hop of nodei and it is the smaller the better. (Hi+1) deals with the root node whose hop is 0. α, β, λ, δ are normalized parameters of these four factors.

$$W_i = \alpha / D_i + \beta / (N_i + 1) + \lambda E_i + \delta / (H_i + 1) \qquad (2)$$

Nodes of network can be divided into three types: root node, sink node and sensor node[10]. Root node is a special node whose energy is unlimited and it is active all the time. Root node is the first network node and at the beginning root node sends broadcast packets to search child nodes. Non-network node saves the broadcast packets and calculates the weight of sink nodes based on Eq. 2. Finally, non-network node selects the best sink node with maximum weight to join the network. If the best sink node refuses the non-network node to join the network, the non-network node needs to select the sub-optimal sink node, third-optimal sink node until it has joined the network successfully. If the non-network node cannot join the network after scanning all available sink nodes, it needs to clear all available sink nodes' information and then saves other broadcast packets to reelect an available sink node. Non-network node begins to select child nodes after joining into the network

B. Adjust Topology Dynamic

In the following two cases we have to reconstruct the network partially.

Case 1. Energy consumption.

The energy consumption of sink node is quicker than sensor nodes and the more child nodes the quicker. After some time of data transmission, if the energy of a sink node reduces to R% we need to delete the farthest child node to balance the energy consumption. Energy percentage R is related to the number of child nodes number and the calculation equation as (2). N is the number of child node. Why we delete the farthest child node: firstly, according to (1) of [8] the farther the more communication energy. Secondly, the farther the smaller probability of reelect this node as sink node.

$$R = N / (N + 1) \qquad (2)$$

Case 2. Link failure.

Child node sends data packets to its sink node periodically and sink node also periodically sends response packets to its child nodes to ensure the links are connected. If a sink node has not received any data packet from a child node in a certain period it judges the link is unsuccessful and removes the child node from its child node table. If a child node has not received any response packet from its sink node in a certain period it will judge the link is unsuccessful and re-select sink node

III. ALGORITHM DESIGN

During the process of network self-organization, nodes are divided into two kinds: network node and non-network node.

All packet types in this paper given in Table.1.

Table 1. Packet type definetion

Symbol	Description
PT_JOIN_REQUEST	Packet of non-network node requests to join in network.
PT_ACCEPTED	Packet from sink node to accept the non-network node joins in network.

PT_DENIED	Packet from sink node to deny the non- network node joins in network.
PT_JOIN_OK	Reply of PT_ACCEPTED.
PT_DELETE	Packet from sink node to delete a child node.
PT_DELETE_OK	Reply of PT_DELETE.
PT_SINK_SEARCH	Broadcast packet of non-network node to search available sink node.

Network Node:

In the network self-organization phase, network node sends some broadcast packets to searching child node actively. After that the node will switch into network monitor state. During this process the node needs to check whether it receives PT_JOIN_REQUEST packet from non-network node and whether the array child[](An array to record information of current child nodes) has available space. If it receives PT_JOIN_REQUEST packets from a non-network node and the array child[] has available space it will send PT_ACCEPTED packet to the non-network node. Otherwise it will send PT_DENIED packet to the non-network node.Then non-network node search for next available maximum weigh of node in sink[ ] (an array to record the weight of neighboring nodes) send PT_JOIN_REQUEST packet again above process is done

Non-network Node:

All non-network nodes are in sleep model before they receive the first packet from network node. If a non-network node receives a packet from network node for searching child node, it will start a timer and saves the information of available sink node in array optional_sink[](An array to record the information of available sink nodes). When the timer is over the non-network node scans array optional_sink[] and selects the best sink node based on (1). Then it sends PT_JOIN_REQUEST packet to the best sink node and waits the reply. If it gets a reply of PT_ACCEPTED it needs to send a PT_JOIN_OK packet to the best sink node. Here this non-network node joins in the network succeed and becomes a network node, it will broadcast packet to search child nodes. If the best sink node replies a PT_DENIED packet that means this non-network node cannot select this best sink node as its sink node.

The non-network node needs to reelect sink node from other available sink nodes until it receives PT_ACCEPTED packet. After scanning all available sink nodes and the non-network node still cannot join in the network, it needs to clear array optional_sink[] and resets timer to wait for searching of other network nodes. All non-network nodes can select a best sink node rapidly. After that they will send packets to request join in network.

Algorithm 1 Select the best sink node

```
1: i ← 0, max_weight ← 0, sink_index ← 0
2: while i < ava_sink_num do
3:     calculate the weight W
4:     optional_sink[i].weight ← W
5:     i + +
6:     if max_weight < W then
7:         sink_index ← i
8:         max_weight ← W
9:     else if max_weight = W then
10:        if optional_sink[i] is greater then
11:            sink_index ← i
12:            max_weight ← W
13:        end if
14:    end if
15: end while
16: Output:sink_index
```

Algorithm 2 Non-network node requests to join in network

```
1: Send PT_JION_REQU EST to node optional_sink[sink_index]
2: Start a timer. The node receives and saves packets during the timer
3: if the node receives a P T ACCEP T ED packet then
4:     is_net_node ← true
5: else if the node received a P T DEN IED packet then
6:     i ← 0, max_weight ← 0, sink_index_tmp ← 0;
7:     while i < ava_sink_num do
8:         if optional_sink[i].weight ≤ optional_sink[sink_index].weight && i≠sink_index then
9:             if optional_sink[i].weight > max_weight then
10:                sink_index_tmp ← i
11:                max_weight ← optional_sink[i].weight
12: else if optional sink[i].weight = max weight then
13:                if optional_sink[i] is greater then
14:                    sink_index_tmp ← i
15:                    max_weight ← optional_sink[i].weight
16:                end if
17:            end if
18:        end if
19:        i + +
20:    end while
21:    if max_weight = 0 then
22:        ava_sink_num ← 0
23:        return
24:    else
25:        sink_index ← sink_index_tmp
26:        go to Line 1
27:    end if
28: end if
```

Step 1. Send PT_JOIN_REQUEST packet to node optional_sink[sink_index] and start a timer.

Step 2. The node receives and saves packets before the timer is over.

Step 2.1. If the node received PT_ACCEPTED packet from its available sink node it needs to send PT_JOIN_OK packet to reply, and then go to Step 4.

Step 2.2. If the node received PT_DENIED packet from its available sink node it needs to reelect a sub-optimal sink node.

Step 3. Reelect a sub-optimal sink node, i=0, max_weight=0, sink_index_tmp=0(A variable to record the index of sink node in array optional_sink[]).

Step 3.1. If i≥ava_sink_num go to Step 3.6.

Step 3.2. If optional_sink[i].weight≤optional_sink[sink_index].weight && i!=sink_index, go on; else go to Step 3.5.

Step 3.3. If optional_sink[i].weight>max_weight, then do sink_index_tmp=i, max_weight=optional_sink[i].weight, and then go to Step 3.5.

Step3.4 If optional_sink[i].weight==max_weight, separate compare the hop, ratio value of residual energy with number of child node, distance of optional_sink[i] and optional_sink[sink_index_temp], sequentially. The priority selection standard is less hops, greater ratio of ((residual energy)/(number of child node+1)), less distances. If the node of optional_sink[i] is better, do sink_index_temp=i, max_weight= optional_sink[i].weight.

Step 3.5. i++, go to Step 3.1.

Step 3.6. If max_weight==0 it means there is no available sink node and the non-network node cannot join the network, then do ava_sink_num=0 and go to End to wait for other nodes' searching; else do sink_index=sink_index_tmp, go to Step 1 to rejoin network.

Step 4. Non-network node joins in the network successful and becomes a network node.

IV. Reorganization of network

When a sink node adds or deletes a child node it will update Eorg[10](The residual energy of last topology change) with Eava(Current residual energy). For prolonging the network lifetime and balancing the energy consumption all sink nodes need to check their energy consumption and Algorithm 2 is the check method.

Algorithm 3. Check energy consumption of sink node

Begin

Step 1. Check whether (Eava/Eorg)≤(R=N/(N+1)) when the energy check timer is over, if (Eava/Eorg)≤(R=N/(N+1)) select the farthest child node and send PT_DELETE packet to it. At the same time update Eorg: Eorg=Eava. Otherwise go to End.

Step 2. After sink node sends PT_DELETE packet it needs to wait for the PT_DELETE_OK packet. If the node receives
PT_DELETE_OK packet it will delete the farthest child node's record and update N: N=N-1.

Step 3. If N==0 the sink node becomes a sensor node, else start a new energy check timer.

End

There is no loop in Algorithm 2 so the complexity is O(1), but after sending a PT_DELETE packet the node starts a timer to wait the reply and it will cost some time. If a node receives a PT_DELETE packet or doesn't receive a reply packet during some time it needs to reelect sink node. Firstly, it broadcasts some PT_SINK_SEARCH packet periodic and start a timer. During the timer, if the node receives reply of PT_SINK_SEARCH packet it will save the sink node's information in array optional_sink[] and update ava_sink_num. Secondly, after the timer it scans the array optional_sink[] to select the best sink node and then join in the network based on Algorithm 1. If the node's hop is changed it needs to inform its child nodes to update their hop.

V. SIMULATION AND ANALYSIS

In this paper all experiments are all based on NS2. Following experiments are all based on self organizing protocol. According to the experiment settings, we set communication radius is 10m, $\alpha = 15$, $\beta = 11$, $\lambda = 1/29$ and $\delta = 11$.transmitting radius of node is 10m

In order to verify the efficiency of self organizing protocol we do five groups experiments. All nodes are randomly distributed in the test area and the number of nodes is 50, 100,150, 200 and 250.
Simulation parameter are taken by below values

parameter	values
Communication radius	15 m
Initail energy	100 j
period of broadcast searching child node message	0.1 s
Send packet power	0.66 w
Transition Time(time taken to state transition from sleep to idle)	0.005s
Receive packet power	0.395 w
Ideal Power	0.02 w
Sleep power	0.035 w

Self organizing protocol has three functions they are self-organize tree-based network, balance energy consumption, reelect sink node the root rode whose coordinates in the border of network,the root node is located at the center of the simulation area.It can be seen that the results of self-organization different due to the difference of root node's location, but every node succeeds in joining the network. the success rate of packet in Self organizing protocol is further higher than LBT , it keeps stable with the number of

sensor nodes increasing.Root node at border The average energy consumption is 0..86*J* in fig.1 more than root node at center . Thus, self organizing protocol can achieves a better performance if the root node is in the center of network.

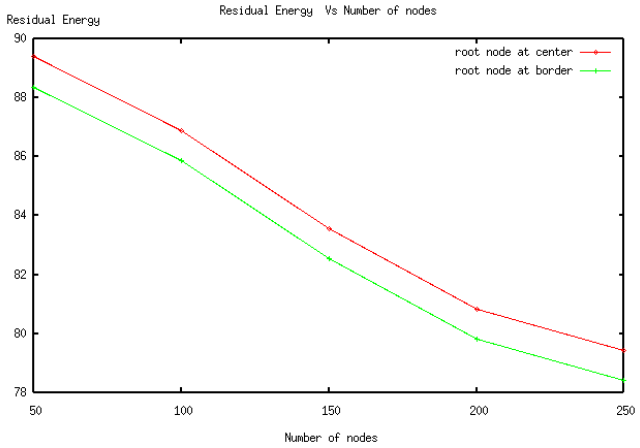

Fig.1.Relationship between number of nodes and residual energy

we evaluate the performance of Efficient Tree-based Self-organizing Protocol , LBT with different number of sensor nodes. The root node is located in the center of topology.It can be seen that the network lifetime of self organizing protocol is longer than LBT, because it periodically checks the residual energy of sink node and re-organize the hot area to achieve energy consumption.What's more, the success rate of packet in self organizing protocol is further higher than LBT because in self organizing protocol using selection of sink node based on the hop count ,number of child node ,distance and using data aggregation to achieve better success rate of packet , it keeps stable with the number of sensor nodes increasing. Thus, the network constructed by self organizing protocol is reliable.

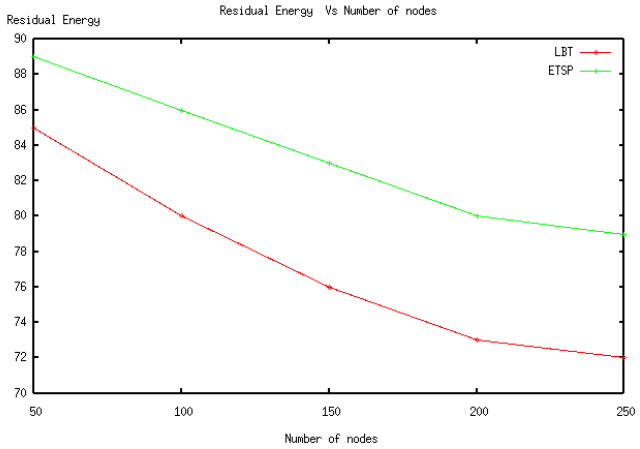

Fig.2.Relationship between residual energy and number of nodes

Residual energy of self organizing protocol is more than than LBT protocol in fig2 . Hop indicates that the selection of sink node is based on the least hop which between the node and available sink node. Distance indicates that the selection of sink node is based on the least distance between the node and available sink node. Left Energy indicates that the selection of sink node is based on the maximum residual energy of available sink node. Child Number indicates that the selection of sink node is based on the number of each child node's available sink node.based on the above process to select the sink node and reorganizing the topology to achieve less energy consumption .Residual energy calculate

Residual Energy= ((total initial energy of nodes-total consumed energy of nodes)/number of nodes)*100

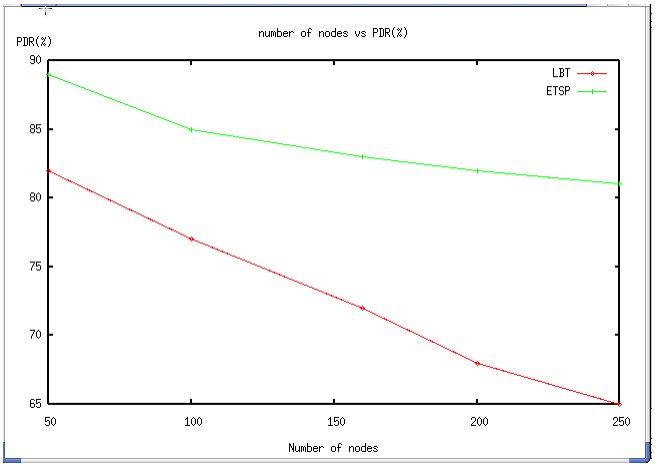

Fig.3.relationship between the packet delivery ratio and number of nodes

In the above fig 3.selft organizing protocol packet deliver ratio is more than the LBT ,her in this paper we using different metrics like selection of sink node based on max weight ,reorganizing and data aggregation this are used to achieve better packet delivery ratio is calculated by below

Packet delivery ratio=(Number of packets receive by nodes/ Number of packet send by nodes)*100

Fig.4. in this Life time of Etsp protocol is more than LBT

Based on the above process we get better network life time

Life time of network can be calculated by below equation: Energy per second = Consum energy/Simulation time(sec) Life time of network = Total initial energy/ Energy per second

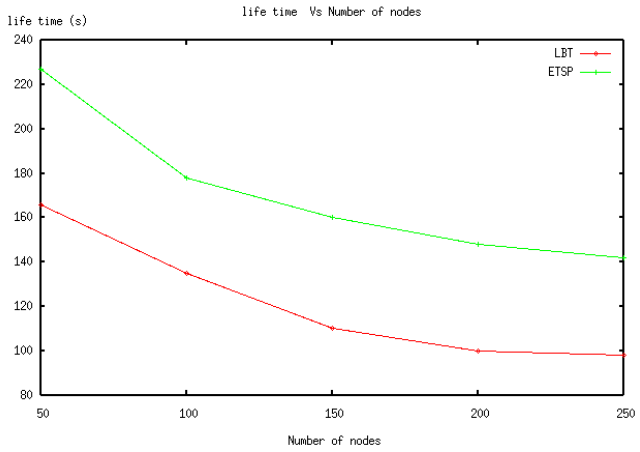

Fig.4. Relationship between life time of network and number of node

VI. CONCLUSION

An efficient self-organization protocol for sensor networks of IoTs.Self organizing protocol saves energy and has a longer packet success rate by constructing a tree-based network quickly. We use the weight of nodes, including residual energy, hop, number of child nodes and distance between the nodes, to determine whether the node can be a sink node.

During the process of data transmission, the network topology changes dynamically. The simulation results show that Self organizing protocol is able to build reliable tree-based networks, reduces the energy consumption and increase success rate of packet

VII. REFERENCES

[1]. Akyildiz IF, Su W, Sankarasubramaniam Y, Cayirci E (2002) Wireless sensor networks: a survey. Comput Netw 38(4):393-422

[2]. Yick J, Mukherjee B, Ghosal D (2008) Wireless sensor network survey. Comput Netw 52(12):2292-2330

[3]. Anastasi G, Conti M, Di Francesco M, Passarella A (2009) Energy conservation in wireless sensor networks: a survey. Ad HocNetworks 7:537-568

[4]. Kwon S, Kim J, Kim C (2008) An efficient tree structure for delay sensitive data gathering in wireless sensor networks. In: 22nd international conference on advanced information network-ing and applications—workshops, 2008. AINAW 2008, pp 738- 743, 25-28 Mar 2008

[5]. Sang Y, Shen H, Inoguchi Y, Tan Y, Xiong N (2006) Secure data aggregation in wireless sensor networks: a survey. In: Seventh international conference on parallel and distributed computing, applications and technologies, pp 315-320

[6]. Lifetime Optimization by Load-Balanced and Energy Efficient Tree in Wireless Sensor Networks

[7]. Khamforoosh K. Clustered balanced minimum spanning tree for routing and energy reduction in wireless sensor networks. IEEE Symposium

on Wireless Technology and Applications 2011; 56-59

[8]. Khamforoosh, K. 2011. Clustered Balanced Minimum Spanning Tree for Routing and Energy Reduction in Wireless Sensor Networks. In Wireless Technology and Applications (ISWTA), 2011 IEEE Symposium on (Sep. 2011), 56-59. IEEE.

[9]. Khan JA, Qureshi HK, Iqbal A. Energy management in wireless sensor networks: A survey. Computers & Electrical Engineering 2015; 41:159-176.

[10]. An Efficient Tree-based Self-Organizing Protocol for Internet of Things ACCESS.2016.2578298, IEEE Access

Cite this article as :

Charan Mangali, "Improving Wireless Sensor Network Lifetime Using Self-Organizing Protocol", International Journal of Scientific Research in Science and Technology (IJSRST), Online ISSN : 2395-602X, Print ISSN : 2395-6011, Volume 6 Issue 4, pp. 330-338, July-August 2019. Available at doi : https://doi.org/10.32628/IJSRST196462
Journal URL : http://ijsrst.com/IJSRST196462