Chapter 6

Assimilation of Correlated Data

6.1 Introduction

Aggregation of data is essentially carried out when large number of nodes have correlated data. Partial computation is done at a node where many sensor nodes send their interrelated data and this aggregated data is forwarded. This in turn reduces repeated transmission of redundant data unnecessarily. Advantage is energy conservation in the network, which is beneficial for long life application. In traditional network, intermediate nodes do not alter the data payload of packets they are forwarding. In this chapter, common part of the header is retained and data is aggregated when information is related, else the information is appended to the existing packet in the case of unrelated data. The packet size is dynamically changed to accommodate the related acquired data. In addition, routing and clustering to assist, in-network aggregation of correlated data is carried out. This process saves the additional computation cost at the base station by carrying out a part of reckoning at the intermediate nodes. Most of the papers deal with the opportunistic aggregation during routing where data handling is not addressed.

6.2 Aggregation Process

The whole area of interest is divided into clusters. Each cluster is such that if there is a node at the center of grid shaped cluster, it is able to communicate to its four neighboring clusters. Node which is approximately at the center of the cluster forms the cluster head. All other nodes associate themselves with the nearest cluster head. Path is established between the cluster heads using the concept of minimum
hop count. Path is established between the non-cluster head nodes such that they send their data to their nearby node. If the data is correlated it is averaged, else it is appended to the existing packet header. Time synchronization to sense, aggregate and forward data are carried out between the cluster head and non-cluster head nodes. The data obtained at the base station thus has all the information of the nodes in the network either averaged or appended.

6.3 Clustering

The aggregation process starts with creation of cluster. Sensor nodes are randomly deployed. In order to avoid multiple unwanted hops, clusters are formed. This assists in aggregating values in a particular region and forwarding it with minimum hop. Each node is in the hearing range of nodes in the neighboring four clusters. Each cluster is in the form of a grid structure.

6.3.1 Creation of clusters

Figure 6.1a shows random deployment of clusters. Figure 6.1b displays how the network is partitioned into small clusters. The arrow in the Figure 6.1b indicates the maximum distance possible between sensor nodes. Every node is having its positional information along with that of the base station. Initially a node at the center of the network is chosen as the base station (node 0 in the Figure 6.1). The cluster is so partitioned that the center of the next cluster is at the maximum transmission range of this cluster center along its four sides. This process is repeated for the whole network. Each cluster is identified by a Cluster_Id. Figure 6.1c depicts the naming of each cluster with its Cluster_Id. e.g. The Cluster_Id 211 has nodes 7 and 56 within its region. The first digit Main Significant Bit (MSB) among the three digits in the Cluster_Id represents one of the four quadrants along the southwest (0), northwest (1), northeast (2) and southeast (3) of the base station. The second digit is signifying the minimum number of hops each node in the cluster is taking towards the base station along the x direction. Similarly the third Least
Significant Bit (LSB) is expressing the hop count along the y direction. According to the Figure 6.1 the base station is in the Cluster_Id 300 with node id 0. Each node in its respective cluster is associates with the Cluster_Id. This is illustrated in Figure 6.1d. Each node computes its distance from the base station. With the knowledge of the transmission range it classifies itself into a specific Cluster.

Algorithm 6.1: Cluster creation and assigning Cluster_Id

Aim: To create cluster in a network with randomly deployed nodes and also to assign a Cluster_Id for each of the regions created.

1. Get the location (x and y coordinates) of the base station.
2. Partition the network such that on four sides of the base station at a distance of half the transmission range of the cluster region is identified (as in Figure 6.1b.) Identify this region as a cluster associated with the base station (in the example Cluster_Id 300).

3. On moving to the left from the border of this Cluster_Id, by an amount equal to the transmission range, next cluster is identified. The second significant digit in the Cluster_Id gives an indication that the cluster region is one hop away along the horizontal from the previous hop. This process is repeated until it reaches the edge of the area of interest.

4. The process is also repeated by moving along the vertical direction by an amount equal to the maximum transmission range. The LSB in the Cluster_Id increments with every hop along the vertical direction. The quadrant to which a node belongs is identified with the MSB. The three digits in the Cluster_Id correspond to the quadrant number, hop along horizontal and hop along vertical directions.

5. The steps 3 and 4 are repeated until the whole network is identified with a Cluster_Id as in Figure 6.1c.

Algorithm 6.2: Node association with a Cluster_Id.

Aim: To associate each of the position aware nodes to a particular cluster region (Cluster_Id) for the whole network.

Each node is aware of its position in the network.

1. Find the distance between the node and the base station.

2. Find the difference between the x coordinates of the individual node with the x coordinate of base station. This assists in finding the MSB as it indicates the quadrant value.

3. Add half the transmission range to the absolute difference obtained from step 2 and divide it with the maximum transmission range to get the second significant digit of the Cluster_Id.
4. Find the difference between the y coordinates of the individual node with 
y coordinate of base station. Getting the value from the step 2 the actual 
quadrant can be identified.

5. Add half the transmission range to the absolute difference obtained from 
step 2 and divide it with the maximum transmission range to get the LSB 
of the Cluster_Id.

6. From step 3, 4 and 5 the Cluster_Id associated with each node is obtained.

7. Update the cluster table of each of the node with the associated 
Cluster_Id.

8. Repeat the steps 1 to 7 to assign Cluster_Id to each of the node in the 
network.

6.3.2 Cluster and Neighbor’s Table

Every node is having a cluster table as shown in Table 6.1, immediately on 
associating it with a particular cluster.

<table>
<thead>
<tr>
<th>Information stored</th>
<th>Size occupied in bytes</th>
<th>Functionality</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cluster_Id</td>
<td>1</td>
<td>Identifies the node with a specific cluster region</td>
</tr>
<tr>
<td>Clust_Dist</td>
<td>2</td>
<td>Distance of the node from the physical center (absolute) of the cluster area (during cluster creation)</td>
</tr>
<tr>
<td>Cluster Head Address</td>
<td>4</td>
<td>Address of the Cluster head for that cluster region</td>
</tr>
<tr>
<td>Cluster Neighbor Address</td>
<td>4</td>
<td>Pointer to the address of the linked list containing neighbors information</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>11 bytes</strong></td>
<td></td>
</tr>
</tbody>
</table>

Computation of the nodes, distance from physical center is stored in the 
Table 6.1 along with the information of the current Cluster Head Address.
Figure 6.2 illustrates the linked list representation of the neighbors which are close to the current node and belonging to the same cluster containing information of their node address and their distance from the physical center. This linked list is an extension of the last entry of Table 6.1. Table 6.2 gives the significance of each field in the neighbors list.

<table>
<thead>
<tr>
<th>Other fields of cluster table</th>
<th>Cluster Neighbor Address</th>
</tr>
</thead>
<tbody>
<tr>
<td>Node Id</td>
<td>Clust Dist</td>
</tr>
</tbody>
</table>

Fig. 6.2: Linked List Representation for the Node’s Neighbor

Table 6.2: Cluster Neighbors Details

<table>
<thead>
<tr>
<th>Information stored</th>
<th>Size occupied in bytes</th>
<th>Functionality</th>
</tr>
</thead>
<tbody>
<tr>
<td>Node Id</td>
<td>4</td>
<td>Identifies the neighboring node’s id</td>
</tr>
<tr>
<td>Clust_Dist</td>
<td>2</td>
<td>Distance of the neighbors distance from the physical center</td>
</tr>
<tr>
<td>Pointer to the next neighbor</td>
<td>4</td>
<td>Link to the next neighbors details</td>
</tr>
<tr>
<td>Total</td>
<td>10 bytes</td>
<td></td>
</tr>
</tbody>
</table>

### 6.3.3 Choice of Cluster Head

Clusters being created, the node which is closest to the physical center of the cluster is chosen as the cluster head. To carry out this process, in each cluster all the nodes is arranged in the increasing order of their distance from the physical center. The node which is closest to the physical center is the cluster head. These nodes identify themselves as cluster head and are updated in their routing table. The choice of cluster head for the current example is shown in Figure 6.3. When the cluster head loses energy then the node next to the current cluster head closest to the center becomes the next cluster head. The control signal used to find the neighbors
belonging to the same Cluster_Id and their distance from the physical center of the cluster region is shown in Figure 6.4. The Packet Type is DIST_COMPUTE. Cluster_Id is the cluster region to which the node belongs to. Clust_Dist is the distance of the node from the physical center of the cluster region. Source Address is the address of the current node. This packet is broadcast to all the neighbors.

![Fig 6.3: Selection of Cluster Head](image)

![Fig. 6.4: Control Packet for finding Cluster Head](image)

**Algorithm 6.3: Choosing cluster head**

**Aim:** To choose a cluster head for each of the cluster region that is identified by the Cluster_Id

1. Absolute physical center in each of the cluster region is identified.
2. In each cluster, the distance between the cluster node and physical center is calculated for each node and stored in Clust_Dist in the cluster table.
3. Update the control packet of the type as shown in the Figure 6.2 with Packet Type as DIST COMPUTE, the Cluster_Id to which it belongs, the Clust_Dist, current node address in Source Address and broadcast it to all its neighbors.

4. If the message reaches neighboring nodes having other Cluster_Id, then the message is freed.

5. All nodes within the cluster update their neighbors list (as in Figure 6.2) with the Node Id (Source Address) and Clust_Dist as obtained from the control signal (Figure 6.4).

6. The neighbor's list is updated such that, they are inserted in the order of increasing distance with respect to (Clust_Dist) from the physical center.

7. The node which is at the beginning of the linked list, form the cluster head.

8. Every node updates their cluster table with the Cluster Head Address.

9. Create routing table in each node and update the Cluster Head Address and Cluster_Id in it.

10. With the current cluster head exhausting its energy, the next node in the linked list forms the next cluster head.

### 6.4 Routing

Routes need to be established between the cluster head to send data to the base station. To accomplish this, routing table is maintained at each of the nodes to direct the flow of packets.

#### 6.4.1 Routing Table

Functionality of each of the elements in the routing table is depicted in the Table 6.3. Int_Hop, Counter and Data Counter are used during the time synchronization phase. Previous Hop Node Address, Next Hop Node Address, Sequence Number and Hop Count are used to establish route between cluster heads. Data Reading, Data Time and Data Pointer are utilized during data packet transmission.
Figure 6.5 illustrates the linked list representation of the data stored in the routing table. The Data Time, Id, Data, Cluster_Id are associated with the node sensing the information. Count is the number of neighboring nodes having correlated value.

Table 6.3: Routing Table

<table>
<thead>
<tr>
<th>Information stored</th>
<th>Size occupied in bytes</th>
<th>Functionality</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cluster Head Address</td>
<td>4</td>
<td>Cluster head address associated with the node</td>
</tr>
<tr>
<td>Cluster_Id</td>
<td>2</td>
<td>The cluster region to which the node belongs</td>
</tr>
<tr>
<td>Near Neighbor Dist</td>
<td>2</td>
<td>Stores the minimum distance among all the neighboring nodes to the current node</td>
</tr>
<tr>
<td>Previous Hop Node Address</td>
<td>4</td>
<td>Address of the node from where data is obtained</td>
</tr>
<tr>
<td>Next Hop Node Address</td>
<td>4</td>
<td>Address of the next hop node where data is going to be sent</td>
</tr>
<tr>
<td>Sequence Number</td>
<td>1</td>
<td>No of times new paths is established</td>
</tr>
<tr>
<td>Hop Count</td>
<td>1</td>
<td>Hops required to reach the destination from the source along the cluster heads</td>
</tr>
<tr>
<td>Int_Hop</td>
<td>1</td>
<td>Maximum number of hops taken so far to reach this node from the source node for any node passing its data through this node.</td>
</tr>
<tr>
<td>Counter</td>
<td>1</td>
<td>Number of nodes incident on this node</td>
</tr>
<tr>
<td>Data Counter</td>
<td>1</td>
<td>Counter which keeps a count of how many data are yet to be received</td>
</tr>
<tr>
<td>Data Pointer</td>
<td>4</td>
<td>Pointer to a list containing the data information from all the previous hop nodes and the current node</td>
</tr>
</tbody>
</table>

Total 25 bytes

![Data Pointer]

Fig: 6.5: The Linked List Representation of the Data Packets in the Routing Table
6.4.2 Routing Packets

Path Establishment between Cluster Heads

To establish a path to the base station from every cluster head, a beacon signal is broadcast from the base station to its neighbors. The path is established based on the minimum hop it takes to reach the base station from the cluster heads. Figure 6.6 shows the packet format for the beacon signals. Packet Type is the BEACON, Sequence Number specifies the number of times the beacons signals are invoked so far. Hop count specifies the number of hops required to reach the base station from the current node.

```
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+------------------------------------------+
| Packet Type | Sequence Number | Hop Count |
| +------------------------------------------|
| Source Address |
| +------------------------------------------+
```

Fig. 6.6: Packet Format for Beacon Signals

Figure 6.7a indicates one hop broadcast signals and Figure 6.7b illustrates dropping of all packets and establishing path only to the first hop cluster head neighbors. Figure 6.7c shows second hop broadcast. Path establishment is depicted in Figure 6.7d. This process repeats to obtain the path as shown in Figure 6.7e. Sum of the middle and the LSB digit of the Cluster_Id specify the minimum number of hop counts required to reach the base station from the current cluster head. Unnecessary broadcasting of the beacon signals during path discovery to the base station can be prevented by avoiding hops beyond this value. This conserves some energy during path discovery process. The knowledge of new quadrant reached during broadcasting also assists in avoiding unwanted broadcast into other quadrants.
Fig 6.7: Stages of Establishing Path between the Cluster Heads (a) One Hop Broadcast (b) Path Establishment Corresponding to this Broadcast (c) Second Hop Broadcast (d) Path Establishment for Second Hop (e) Path Establishment for the Whole Network

Algorithm 6.4: Path establishment between cluster head

Aim: To establish a path between cluster head using minimum hop from the base station to the source node

1. The control packet shown in Figure 6.6 is broadcast from the base station to its one hop neighbors. The Sequence Number is initially set to 1 and the Hop Count initialized to zero. The Sequence Number is incremented with every new path discovery process.

2. On reaching the next node, the Hop Count in the control packet is incremented. Nodes receiving this control packet carrying the information of Sequence Number and Hop Count will check if there is already a path of
lower Hop Count or same Hop Count with same Sequence Number available in the routing table. If so, free the packet.

3. If the Sequence Number in the control packet is higher than that in the routing table, update the routing table with
   a) The Next Hop Node Address with the value of the address from where the control packet is obtained. (The path to be established is towards the base station).
   b) The Sequence Number
   c) Hop Count

4. If the new control message has same Sequence Number as that of the Sequence number in routing table and the new message has a lower Hop Count from the base station to the source node, update in the routing table
   a) The Next Hop Node Address with the value of the node address from where the control packet is obtained.
   b) Hop Count

5. Steps 2 to 4 are repeated until it able to broadcast and reach as many nodes as possible in the whole network.

Path Establishment among Nodes within Clusters

Nodes (Source) in each of the clusters broadcast a one hop message of the type in Figure 6.8 to all its neighbors. The Packet Type is NEXT_HOP_SEARCH. The Source Node’s x and y coordinate along with Source Address which initiated this process is broadcast to the neighbors. The node which is physically far from the center associates with a neighbor which is closer towards the center. Once a particular node is associated with any node, it is not considered for association any further. This node associates with its next neighbor towards the cluster head.
Figure 6.9a shows a part of the topology. The broadcast message is sent to neighboring node as is presented in Figure 6.9b. Figure 6.9c illustrates the path establishment between the non-cluster head nodes towards the clusterhead. Reasoning logically for each cluster head, the maximum number of input that is possible from within the cluster is four. Figure 6.10 shows the path establishment between non-cluster head nodes in the current topology.

Fig. 6.9: Stages during Establishment of Path between Nodes within the Cluster
(a) Topology (b) Broadcast Control Signal (c) Establishing Path

Fig. 6.10: Establishing Path for the Whole Network
Algorithm 6.5: Path establishment among nodes within the cluster region

Aim: To establish a path between the non-cluster head nodes to its nearest neighbor

1. The Next Hop Node Address is set to the Cluster Head Address by default. (Initialization Process). Clust_Dist (The distance between the current node and its cluster head) is calculated and stored in the cluster table. Near Neighbor Dist (stores the minimum distance among all the neighboring nodes information as obtained from the control field so far) is initialized to value stored in Clust_Dist initially.

2. The control signal of the type as in Figure 6.8 is broadcast to all the neighboring nodes in the network from all the non-cluster head nodes. This number is incremented with every new path that is established. This value is updated in the routing table.

3. If the neighbors which receive control signal belong to some other cluster region (id), free the control packet. Go to Step 8.

4. Source Node (non-cluster head node) address along with its x and y coordinate is obtained for packets belonging to the same cluster.

5. Two distances are computed from the coordinates of the Source Node. One is the distance of the Source (neighboring) Node from the cluster head (stored as the Clust_Dist in the neighbors list of Figure 6.2) and other the distance between the current node and the Source Node.

6. If the distance between the Source Node to this node is larger than the Clust_Dist (distance between the current node and the cluster head) free the packet. (This indicates that the neighboring node is far away from the cluster head or in the other quadrant of the same cluster. This also ensures that path is established only from node which is far away to node which is nearby to the cluster head).

7. If the distance between current node and the neighbor node is less than Near Neighbor Dist (minimum distance among neighbors of the current
nodes obtained so far), the Source Address becomes the Next Hop Node Address. *Free* the packet.

8. The steps 3 to 7 are repeated for all the control signals received by each of the nodes.

If all the neighbors are farther than the distance to the cluster head, the cluster head becomes the Next Hop Node Address.

### 6.5 Time Synchronization

Sensor nodes read the data, and send their data to the neighboring nodes as specified in the routing table. The data is moved from one node to another in a multi-hop manner until it reaches the cluster head and eventually from these cluster heads to the base station. The time instance when the data is read by various sensors may not synchronize with each other. In order to achieve forwarding of recently read data a two-step procedure is followed, synchronizing within the cluster region and synchronization among the cluster heads.

#### 6.5.1 Synchronization between Neighbors of Non-Cluster Head Nodes

Nodes in each of the clusters are keeping a count of number of nodes sending data to it (in addition to a count of one about it-self). This gives a count of number of data each node is expected to receive. Figure 6.11 shows the count of data expected at each node. When data transmission is taking place, this count is decremented with each data received. As soon as the count reaches zero aggregated data is moved to the next node. The count may not always reach zero if some of the packets are dropped. In such cases, the data is forwarded after waiting for a specific period (equal to time required to send data by one hop x number of nodes yet to send data on it). If data arrives within this waiting period it is forwarded along with other
data on aggregation. If the data arrives at a later interval, that data is simply forwarded without delay.

![Diagram](image)

**Fig. 6.11: Updation of the Value of Data Counter**

**Algorithm 6.6: Computing the number of nodes incident at a Node**

Aim: To find a count of how many data are to be received by each of the nodes.

1. Counter field in the routing table is all set to one corresponding to the one data read by the current node.

2. A control packet with only Packet Type as NODE_COUNT is sent from all the nodes to reach the base station.

3. As it reaches Next Hop Node Address as per the routing table the Counter field is incremented by one and the packet is freed.

4. Counter field indicates the number of nodes incident at each of the nodes.
Algorithm 6.7: Time synchronization within the cluster

Aim: To obtain the time synchronization between receiving and forwarding data up to the cluster head from the non-cluster head nodes.

1. Each of the nodes set their Data Counter value in the routing table with the value of the Counter field.
2. During the process of data transmission when a data is received in the node, Data Counter field is decremented by one. This gives a count of number of neighbor's incident on this node.
3. As soon as the Data Counter field reaches zero, data is forwarded to the Next Hop Node Address.
4. If the Data Counter field does not reach zero, the node waits for a period equal to the Int_Hop x time to receive data from one hop neighbor. Data is forwarded to the Next Hop Node Address after the waiting period.

6.5.2 Synchronization between Cluster Head

Data information present at the cluster head is transmitted along the cluster heads to reach the base station. Each cluster head has a counter called Int_Hop which maintains the maximum hop count taken by any node to reach this node; when data is to be forwarded to the base station passing through this node. Data from each cluster head data is not passed on to the Next Hop Node Address until a time interval (of time period for single hop x Int_Hop) is elapsed. This helps all the data to reach the cluster head. With each data reaching the cluster head, the Data Counter is decremented and data is forwarded. Figure 6.12 shows the process of finding the maximum hop count.

Figure 6.13 shows the packet format used to find the waiting time meant for synchronization. The Packet Type is HOP_INFO.
Algorithm 6.8: Maximum Hop Count computation

Aim: To compute the maximum hop count required to reach intermediate nodes from the source node; along the path when data is to be sent to the destination.

1. The Int_Hop specifies the maximum number of permissible intermediate node hop count required to reach this node (initially set to zero for all the nodes).
2. The packet format as shown in Figure 6.13 is *unicast* from every node as per the Next Hop Node Address present in the routing table to reach the base station.
3. The Int_Hop field is incremented with every hop.
4. The routing table is updated with the Int_Hop value from the control packet if the value carried in the control packet is *higher* than the value present in
the routing table; it then forwards the packet to the Next Hop Node Address as per the routing table.

5. If the value of Int_Hop is less than the value in the routing table, free the packet.

6. Repeat steps 3 to 5 until it reaches the base station.

Algorithm 6.9: Time synchronization between cluster head

Aim: To obtain the time synchronization between receiving and forwarding data between the cluster head.

1. Each node in the cluster head is having the Data Counter value same as the Counter field for every new interval of data transmission.

2. At each cluster head node, the Data Counter field is decremented by one with each data arrival.

3. If the Data Counter field reaches zero, data is forwarded to the Next Hop Node Address.

4. If the Data Counter field has not reached zero; data is only forwarded after a waiting period (Int_Hop x time for one hop) is elapsed.

5. If the packets are missed out (not able to reach cluster head), they can report to the Next Hop Node Address at a later interval. In the process it could also catch up with the already transmitted data packet if it has not reached the base station.

6.6 Dynamic Data Packet Management

Data is sent from one cluster head to another as per the Next Hop Node Address. Latency and energy consumption increase with the increase in the number of retransmissions in a wireless medium. Control packet overhead increases linearly with node density. Many sensors report their data at approximately the same time and have similar headers; considerable savings can be realized by combining different packets into one large packet with a single header. If the data is related
bearing tolerable accuracy, the value is summed and appended to the data packet along with a counter indicating the number of related values. At the cluster head the value is aggregated. The packet size changes dynamically with the data. Correlated data is averaged and appended to the existing packet destined to the base station.

### 6.6.1 Data Packet

The packet format for sending data is as shown in the Figure 6.14. Dynamically the packet size is changing based on the data volume. When the data is transferred from one node to another the common header is having the information pertaining to various layers. If the data is sent individually to the base station then there is a lot of processing at the base station. The number of packets sent is also large. There is a likely hood of lot of collision and dropping of packets. To avoid this, packet is aggregated at every node as it traverses. This reduces the traffic and also the number of collisions. On avoiding transmission of common data header multiple times, traffic volume is reduced. This is done on appending the data information to the end of the packet header. The size of the data packet is varied dynamically. The functionality of each of the field in the packet is specified in the Table 6.4.

<table>
<thead>
<tr>
<th>Information stored</th>
<th>Size occupied in bytes</th>
<th>Functionality</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Time</td>
<td>4</td>
<td>Time instance when the data was sensed or read</td>
</tr>
<tr>
<td>Id</td>
<td>4</td>
<td>Id of the node reading the data</td>
</tr>
<tr>
<td>Data</td>
<td>2</td>
<td>Data read at that instance</td>
</tr>
<tr>
<td>Count</td>
<td>1</td>
<td>A count of number of data in the neighborhood having the same data with acceptable accuracy</td>
</tr>
<tr>
<td>Cluster_Id</td>
<td>1</td>
<td>The cluster region to which the data belongs</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>12 bytes</strong></td>
<td></td>
</tr>
</tbody>
</table>
6.6.2 Packet Handling

Sensors sense the data at varying intervals of time. In some the data is sensed and aggregated locally before receiving the data from the previous hop node.

Algorithm 6.10: Packet Handling

A1) Aim: To handle data packets arriving at different intervals of time within a cluster.

On receiving a data packet within the cluster,

1. With each new interval of transmitting the data, the Data Counter value is set to the value of the Counter field (both fields are in the routing table). The Data Counter field is *decremented* by one with each data packet received.

2. If the node receiving the data packet is a cluster head go to Step 6.

3. If the node receiving the data packet is *not* a cluster head and if the Data (as obtained from data packet in the current node) is the *fresh* data (Data Time from data packet compared with CURRENT_TIME helps in identifying freshness of data) the following steps are carried out.

4. If the node is not preceded by another node (Counter field in the routing table is one in the routing table), the data packet is forwarded to the Next Hop Node Address as per the routing table. Go to Step 10.

5. If the node is an intermediate node (Counter field is greater than one) with data about itself, or from previous node (Id obtained in the data packet when compared with the current node Id will suggest if it is a data packet
of the current node or from the previous node) with or without data packet received so far from any other previous node,

a) If the Data Pointer is Null, a memory location to store the Data Time, Data field, Id, Cluster_Id and Count are allocated in the node as per Figure 6.5 and updated with values from the corresponding data packet fields and Count field initialized to one. This list is inserted into the list and the Data Pointer points to this entry. Go to Step 6.

b) If the received Data from the data packet matches by an amount equal to the threshold value; with the any of the Data as pointed by the Data Pointer in the listed entries in the routing table for the current Cluster_Id; increment the Count Field corresponding to that Data (value) field in the listed entry. The new Data is summed up with the existing Data in the listed entry and replaces the earlier Data entry with this new sum.

Else

A memory location to store the Data Time, Data field, Id, Cluster_Id and Count is allocated as per Figure 6.5 and updated with values from the corresponding data packet fields and Count field initialized to one. Append this data part to the end of the listed entry as pointed by the Data Pointer in the routing table.

6. If the current node is a cluster head, and the Data Counter is zero or waiting time is elapsed, with Data Pointer not null, for each Data received in the listed entry in the routing table, use the Count field to average the value of the Data and store it back into the Data field for the current cluster region.

7. The pointer to data information part of the data packet is the Data Pointer part present in the routing table with the format as in Figure 6.14.

8. If the Data Counter is greater than one, (packets are yet to be received by the node) free the received packet.
9. If the value of the Data Counter is set to zero, data is immediately forwarded to the Next Hop Node Address.

   Else

   Wait for a time interval equal to the Int_Hop x Time required to send data with one Hop. (Repeat from step 2 for any new data arrived within this interval). After this waiting period, forward the data to the Next Hop Node Address as in the routing table.

10. If the current node is a cluster head go to current algorithm 6.10 A2 Step 2 which follows.

11. Refresh the routing table.

A2) Aim: To handle data packets arriving at different intervals of time across the cluster.

If the node is a cluster head

1. Decrement the Data Counter by one.

2. If the node is a base station. Log the data into the database system to take necessary action.

3. If the node is an intermediate cluster head node (Counter field is greater than one) with fresh data obtained in the current node.
   a) If the previous hop node is not a cluster head follow the steps in the current algorithm 6.10 A1 Step 5.
   b) If the previous hop node is a cluster head belonging to some other cluster, append the data part of the packet to listed entry as pointed by Data Pointer of the routing table.

4. Free the received packet if the Data Counter is greater than one. The pointer to data information part of the data packet is the listed entry as pointed by the Data Pointer in the routing table with the format as in Figure 6.14.

5. If the value of the Data Counter is set to zero, data is immediately forwarded to the Next Hop Node Address.

   Else
Wait for a time interval equal to the Int_Hop x Time required to send data with one Hop. After this waiting period, forward the data to the Next Hop Node Address as in the routing table. (Any packet received during this interval repeats the Steps 1 of current algorithm 6.10 A2.

6. Refresh the routing table. Repeat from Step 1 to Step 5 of algorithm 6.10 A2 until it reaches the base station.

### 6.7 Simulation Results

Data transmitted by identical sensors spaced closely together tend to be spatially correlated. Compression technique is used to remove the redundancy and minimize the amount of traffic in the network.

Ns2 is used for carrying out the simulation. The soil moisture content is noted once in every 5 minutes. The input sensor data is stored in a file. The rate of data acquisition changes based on the rate of variation in the data. If the data is changing rapidly, then the rate of data acquisition occurs fast. The interval of acquiring the data is once in every 150 seconds, 300 seconds or 600 seconds. The reading is not noted during the night times for convenience. Table 6.5 shows the input data read at a particular node. Similar data are read by all the sensor nodes in the network. The output is shown in the Table 6.6 which is obtained at the base station. Figure 6.11 is the topology for which the reading is obtained in the table.

<table>
<thead>
<tr>
<th>Time period in seconds</th>
<th>Data reading at a particular node</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>2.096</td>
</tr>
<tr>
<td>300</td>
<td>2.098</td>
</tr>
<tr>
<td>600</td>
<td>2.099</td>
</tr>
<tr>
<td>750</td>
<td>2.100</td>
</tr>
<tr>
<td>900</td>
<td>2.140</td>
</tr>
<tr>
<td>1500</td>
<td>2.106</td>
</tr>
<tr>
<td>2100</td>
<td>2.107</td>
</tr>
<tr>
<td>2250</td>
<td>2.110</td>
</tr>
</tbody>
</table>
The Current Time refers to the time period when the data packet is obtained at the base station. Data Time is the time when the data is sensed. Id is the node id having the Data reading and number of its neighbor having related data is in Count Column along with the Cluster_Id to which the node belongs. If the data is beyond the threshold value (currently taken as 0.001) among the neighboring node, then its Id, with the Data, Data Time, and Cluster_Id along with Count is appended to the data packet. If the data is matching then the Count Field is increased representing those many nodes carrying related data. If the neighboring data is not able to send data in time to the Next Hop Node Address, then it is sent later, as in the case of node 28 sending data at the Current Time 10.16 seconds as against the earlier sent Current Time of 9.5 seconds when it is supposed to be sent along with node id 27.

The result of the simulation is displayed in the Table 6.7. Considering row I, there is one (Count Field) which has a reading of 2.094 (Data) which is sensed at the time 7.50 (Data time) by a node with Id 51 present in the cluster with Id 300. This is forwarded to a node with Id 32 and it has a data of 2.096. This data is again forwarded to the node with Id 24 and there are two (Count) nodes which have the correlated data of 2.102. This Data is obtained after averaging the two values. Data information of all the nodes is displayed. If two nodes belonging to some other cluster have related reading, it is not aggregated. It is found that the direct transmission suffers due to dropping of packets because of heavy traffic. At the base station the data is obtained from its neighbors 1,2,3,4 and node 24. In the case of aggregation only five packets reach the destination. In the current example seven packets reach the destination. Packet from cluster 112 is sent twice because of the delay caused in sending the Data by Node 28 (Set I Current Time 10.16) and Node 42 (Set II Current Time 10.16). It should have been sent along with Node 27 (Set VI at Current Time 9.5). If each of the nodes were to send the data 74 packets are supposed to reach the destination. With time, the dropping rate of packet increases in the case of direct transmission. In the case of aggregated transmission multiple transmissions increases (as in Id 27, Id 28 and Id 42 in Table 6.7). Multiple
transmissions are caused because of mismatch in the time synchronization during data sensing, aggregation and transmission between nodes along the path.

Table 6.6: Sample Output at the Base Station

<table>
<thead>
<tr>
<th>Current Time</th>
<th>Data Time</th>
<th>Id</th>
<th>Data</th>
<th>Count</th>
<th>Cluster Id</th>
<th>Data Time</th>
<th>Id</th>
<th>Data</th>
<th>Count</th>
<th>Cluster Id</th>
<th>Data Time</th>
<th>Id</th>
<th>Data</th>
<th>Count</th>
<th>Cluster Id</th>
</tr>
</thead>
<tbody>
<tr>
<td>8.60</td>
<td>7.50</td>
<td>51</td>
<td>2.094</td>
<td>1</td>
<td>300</td>
<td>7.50</td>
<td>32</td>
<td>2.096</td>
<td>1</td>
<td>300</td>
<td>7.50</td>
<td>24</td>
<td>2.102</td>
<td>2</td>
<td>300</td>
</tr>
<tr>
<td>8.83</td>
<td>7.50</td>
<td>72</td>
<td>2.093</td>
<td>2</td>
<td>301</td>
<td>7.50</td>
<td>48</td>
<td>2.096</td>
<td>4</td>
<td>301</td>
<td>7.75</td>
<td>17</td>
<td>2.097</td>
<td>2</td>
<td>302</td>
</tr>
<tr>
<td>8.95</td>
<td>7.50</td>
<td>20</td>
<td>2.100</td>
<td>1</td>
<td>220</td>
<td>7.50</td>
<td>49</td>
<td>2.098</td>
<td>5</td>
<td>210</td>
<td>7.75</td>
<td>41</td>
<td>2.096</td>
<td>3</td>
<td>312</td>
</tr>
<tr>
<td>9.12</td>
<td>7.85</td>
<td>31</td>
<td>2.088</td>
<td>1</td>
<td>22</td>
<td>7.80</td>
<td>13</td>
<td>2.102</td>
<td>1</td>
<td>12</td>
<td>7.50</td>
<td>35</td>
<td>2.95</td>
<td>6</td>
<td>11</td>
</tr>
<tr>
<td>9.50</td>
<td>7.50</td>
<td>22</td>
<td>2.95</td>
<td>2</td>
<td>10</td>
<td>7.50</td>
<td>52</td>
<td>2.092</td>
<td>2</td>
<td>21</td>
<td>7.75</td>
<td>55</td>
<td>2.097</td>
<td>2</td>
<td>121</td>
</tr>
<tr>
<td>9.50</td>
<td>7.50</td>
<td>25</td>
<td>2.80</td>
<td>3</td>
<td>20</td>
<td>7.50</td>
<td>65</td>
<td>2.096</td>
<td>1</td>
<td>122</td>
<td>7.75</td>
<td>27</td>
<td>2.097</td>
<td>2</td>
<td>112</td>
</tr>
<tr>
<td>9.50</td>
<td>7.50</td>
<td>59</td>
<td>2.95</td>
<td>4</td>
<td>111</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10.16</td>
<td>7.50</td>
<td>28</td>
<td>2.100</td>
<td>1</td>
<td>112</td>
<td>7.50</td>
<td>42</td>
<td>2.098</td>
<td>1</td>
<td>112</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10.20</td>
<td>7.50</td>
<td>40</td>
<td>2.100</td>
<td>2</td>
<td>101</td>
<td>7.50</td>
<td>7</td>
<td>2.098</td>
<td>2</td>
<td>211</td>
<td>7.75</td>
<td>39</td>
<td>2.096</td>
<td>3</td>
<td>221</td>
</tr>
<tr>
<td>7.70</td>
<td>66</td>
<td>2.092</td>
<td>1</td>
<td>103</td>
<td>7.75</td>
<td>61</td>
<td>2.094</td>
<td>4</td>
<td>102</td>
<td>7.56</td>
<td>33</td>
<td>2.094</td>
<td>2</td>
<td>213</td>
<td></td>
</tr>
<tr>
<td>7.50</td>
<td>74</td>
<td>2.95</td>
<td>3</td>
<td>212</td>
<td>7.50</td>
<td>64</td>
<td>2.092</td>
<td>3</td>
<td>222</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>609.17</td>
<td>607.5</td>
<td>51</td>
<td>2.095</td>
<td>1</td>
<td>300</td>
<td>607.5</td>
<td>32</td>
<td>2.097</td>
<td>1</td>
<td>300</td>
<td>607.5</td>
<td>24</td>
<td>2.098</td>
<td>2</td>
<td>300</td>
</tr>
<tr>
<td>612.21</td>
<td>607.5</td>
<td>64</td>
<td>2.099</td>
<td>1</td>
<td>222</td>
<td>607.5</td>
<td>62</td>
<td>2.098</td>
<td>1</td>
<td>222</td>
<td>607.5</td>
<td>60</td>
<td>2.97</td>
<td>1</td>
<td>221</td>
</tr>
<tr>
<td>607.62</td>
<td>15</td>
<td>2.098</td>
<td>2</td>
<td>212</td>
<td>607.78</td>
<td>7</td>
<td>2.098</td>
<td>2</td>
<td>211</td>
<td>607.8</td>
<td>2</td>
<td>2.098</td>
<td>2</td>
<td>101</td>
<td></td>
</tr>
</tbody>
</table>

Table 6.7: Simulation Parameters Specific to a Sample Input

<table>
<thead>
<tr>
<th>Topology</th>
<th>Random deployment</th>
</tr>
</thead>
<tbody>
<tr>
<td>Network size</td>
<td>50 nodes</td>
</tr>
<tr>
<td>Data acquisition interval</td>
<td>150</td>
</tr>
<tr>
<td>Simulation Period</td>
<td>13 hours</td>
</tr>
</tbody>
</table>

The parameter used for the conduction of the experiment is shown in the Table 6.7. The computation of the simulation result is carried out for the sample input as shown in the Figure 6.15. The topology uses 50 nodes randomly deployed across the
network. Results of the simulation are shown in the Table 6.8. AODV works similar to B_AODV.

![Sample Input Topology](image)

**Fig 6.15: Sample Input Topology**

**Table 6.8: Comparison of Energy Consumption between AODV and Correlated Data Aggregation and Transmission (COAGG)**

<table>
<thead>
<tr>
<th>Network Area</th>
<th>Square Topology. 25 nodes placed at a distance of 200 meters between them. Another 25 nodes dispersed randomly</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sampling Interval</td>
<td>600 seconds</td>
</tr>
<tr>
<td>Routing Protocol</td>
<td>AODV with 802.15.4</td>
</tr>
<tr>
<td>Average Energy (J)</td>
<td>0.0986</td>
</tr>
<tr>
<td>Routing Packets</td>
<td>1517</td>
</tr>
<tr>
<td>Routing Load</td>
<td>32.276</td>
</tr>
<tr>
<td>Packet Delivery Ratio (%)</td>
<td>47/49 x100 = 95.91</td>
</tr>
<tr>
<td>Packets sent</td>
<td>49</td>
</tr>
<tr>
<td>Packets received</td>
<td>47</td>
</tr>
</tbody>
</table>

**Inference from Table 6.8:**

a) Energy Consumption: Consumption of energy is reduced in the case of Correlated Aggregation. This is because the number of packets reaching
the base station is reduced. It is also observed that it also avoids repeated transmission of the header which is common to all the packets. (In the last column 6409.19 J of energy is consumed by correlated aggregation whereas 6418.67 J is used by AODV using IEEE 802.11 and 5823.66 J using AODV with IEEE 802.15.4). Heavy data rate is not supported by IEEE 802.15.4 and hence there is a drop in the delivery of the packets.

b) Routing Packet: Routing packets have reduced in the case of correlated aggregation. This is because many packets are aggregated and sent collectively. (In the last column 274777 packets are sent for 80.8 % delivery ratio as compared to required 482567 packets required for 99.8 % delivery ratio).

c) Packet Delivery Ratio with respect to dropping of packets: The packet delivery ratio is not as good as in IEEE 802.11 using AODV, but it is better than sending all the data to the base station using IEEE 802.15.4 using AODV. Nearly 80.8% of the packets reach the destination as compared to 99.8% using AODV with 802.11

d) Packet Delivery Ratio with respect to actual data received: Packet Delivery Ratio is better than using IEEE 802.15.4 with AODV. This is due to the reduced number of packets which are sent to the base station aggregating data along the path. (The data packets received at the base station corresponding to 300 second interval along the last row shows that the number of packets received is 5. This number suggests that these packets are received from the four nodes close to the base station. The additional one is obtained with because of some delay. Each of these packets has data information pertaining to the area they are aggregating along the path).

Simulation is also carried out for other topology with similar appreciable results.
6.8 Summary

The current chapter focusses on routing and aggregation and providing synchronization during packet transmission. Routing is carried out such that correlated data are sent to neighboring non-cluster head nodes. This assists in aggregation of data at each node when packet is forwarded. Un-related data is appended to the end of the existing packet and forwarded. This results in dynamic size of the packet. The common part of the header is retained and only the data part is appended to the common header along the path. This also avoids repeated transmission of common part of the header and helps in conserving energy. Data between cluster head follow minimum hop transmission to reach the base station. There is a considerable improvement in the energy saving with this approach. Traffic is reduced. Collision and dropping of packets are avoided. The power consumption is reduced by reducing the number and size of the communicated data. This in turn extends the lifetime of the sensor nodes instead of sending the data directly to the base station.

The algorithm could be improved by incorporating paths such that the aggregating nodes have a balanced load so that the consumption of energy is uniformly maintained.