International Journal of Electronics Communication and Computer Engineering
Volume 8, Issue 3, ISSN (Online): 2249–071X

Energy Efficient Adaptive Storage Node Placement for Data Dissemination in Wireless Sensor Networks

Vipan Arora[*]
Research Scholar, NIT, Hamirpur (H.P.), India.

Dr. T.P. Sharma
Associate Professor, NIT Hamirpur (H.P.), India.

[*]Corresponding author email id: vipan.arora@gmail.com

Date of publication (dd/mm/yyyy): 23/05/2017

***Abstract* — Wireless sensor networks (WSNs) have a limited energy supply and limited bandwidth available. Since radio communication is expensive in terms of energy consumption, the sensor nodes typically spend most of their energy reserve on wireless communication (rather than on CPU processing) during data dissemination and retrieval. Storing data in-network at specific locations within WSNs that minimize data transmissions reduces the energy consumption, and hence extends its lifetime. However, finding those suitable storage nodes for data storage becomes a fundamental problem. In this paper, we purpose Energy Efficient Adaptive Storage Node Placement for Data Dissemination (EEASNPDD) in Wireless Sensor Networks to minimize the energy consumption in sensor networks. Our goal is to prolong the network life time by finding optimal storage locations for data so as to minimize the data/query traversal path and thus to minimize total energy cost in data accumulation, transmission and data querying. To achieve this, we purpose to form two storage zones to store the sensed data and to process the query from sink. The scheme finds the optimal storage location using the DV-HOP and PSO algorithm for sensor zones based on data rate, query rate and number of hops. Simulation results show that EEASNPDD provides substantial energy saving as compared to existing schemes.**

***Keywords* — Sensor Nodes, Wireless Sensor Network, Storage Nodes, Data Rate, Query Rate.**

I. Introduction

Wireless Sensor Network (WSN) [1] [2] consists of large number of SNs which are small in size, low cost and have limited memory, sensing, computation and wireless communication capabilities. SNs measure the ambient conditions from the environment surrounding them. The applications of WSNs vary from health monitoring to battle field surveillance to battlefield. In these applications, SNs are deployed to operate autonomously in unattended environments. Since these sensing devices have limited power reserves, therefore WSN lifetime is small. To extend network lifetime energy consumed by these nodes must be optimized. The energy consumed by a SN for communication is much higher than that for computation. Hence, energy consumption in a WSN for data retrieval and delivery needs to be minimized. Besides energy, bandwidth available for communication is another scarce resource [3].

There are two approaches to data handling in WSNs. In first approach, a WSN can be left to its bare functions of data collection. All data is then forwarded to a base station or sink for data analysis. In such a case, there is little computational requirement from the processing unit on the SNs. This results in large amounts of data streams traversing through the network, leading to excessive power drain, a critical resource in WSNs. Second approach is to leave the collected data on certain SNs, while retrieving specific needed data by running queries through the network. This inhibits the raw data streams generated at SNs from traversing towards data collectors (i.e. sinks), instead converted to a problem of limited data extraction [4]. Organizing data dissemination paths with some nodes acting as data store leads to a more efficient use of bandwidth and energy supply.

Data storage has emerged as an effective way in improving network performance by reducing network traffic, alleviating load on sink nodes and decreasing access latency. The collected data can either be stored in the network sensors or transmitted to the sink. Several problems arise when data are stored in sensors. First, a sensor is equipped with only limited memory or storage space, which prohibits the storage of a large amount of data accumulated for months or years. Second, since sensors are battery operated, the stored data will be lost after the sensors are depleted of power. Third, searching for the data of interest in a widely scattered network field is a hard problem. Alternatively, data can be transmitted back to the sink and stored there for future retrieval. This scheme is ideal since data are stored in a central place for permanent access. However, the sensor network's per node communication capability is very limited [5], [6].A large amount of data cannot be transmitted from the sensor network to the sink efficiently. Furthermore, the data communication from the sensors to the sink may take long routes consuming much energy and depleting of the sensor battery power quickly. In particular, the sensors around the sink are generally highly used and exhausted easily, thus, the network may be partitioned rapidly.

Query is used to provide information about environment sensed by SNs to end .A user query may take various forms, e.g., "What is the current position of truck?" and "what is the average temperature of the sensing field?"In this scenario, each sensor, in addition to sensing the nearby environment, is also involved in routing data for two network services: the raw data transmission to storage nodes and the transmission for query diffusion and query reply. There are two possible ways for sensed data. First is transmitting all the data to the sink and second is storing them on each SN locally. Data solely stored in the sink is beneficial to the query reply incurring no transmission cost, but the data accumulation to the sink is very costly. Even if sink is capable enough to at least receive and transmit all individual readings to base station, it soon drains its energy source. Moreover, not each individual reading is required at base station. Ideally, base station issues queries based on application (control or analysis) running there and sink serves these queries by further querying selected readings from recent past. Therefore, the issue here is to store these huge number of sensor readings generated in a given time window somewhere in WSN. Obvious choice seems to store readings at nodes on data/query path from source SNs to sink. Storing data locally incurs zero cost for data accumulation, whereas the query cost becomes large because a query has to be diffused to the whole network and each sensor has to respond to the query by transmitting data to the sink. Data storage strategy is critical in a WSN [7]. The storage nodes not only provide permanent but also serve as a buffer between the sink and the SNs. The positioning of storage nodes, however, is extremely important in this communication model. A bad placement strategy may waste the storage resources and have an adverse effect on the performance. Therefore, a good algorithm for placing storage nodes is needed to strike a balance between these two extremes characterizing a trade-off between data accumulation and data query.

The storage of data in sensor networks is dependent on two main factors: First is data rate i.e. data sensing rate of source nodes and query rate of sink node. Sensing rate of source nodes is the rate at which SNs sense the event, generate sensed data and disseminate it towards storage nodes/sink node. Query rate is the rate at which sink node issues queries towards storage nodes/source nodes. The second is the path distance to the storage nodes. These two factors impact data storage-related communication cost in sensor networks. If the data rate is more than query rate, which means more and more sensed data is generated. The storage zones should be near to the source nodes as this will reduce the communication cost between sensed data and storage zones. If the query rate is more than data rate, then storage zones should be near the sink nodes thereby reducing the overall communication. If the data rate and query rate are same, then storage zones should be at the center of the network. Closer the storage zones to the source and sink, shorter the hop distance, the cheaper it is to store and query a fixed quantity of data. The optimal strategy is to place storage nodes adaptively according to data and query rate so that the communication cost is minimal.

Hence, in this chapter we propose Energy Efficient Adaptive Storage Node Placement for Data Dissemination (EEASNPDD) scheme to identify the optimal locations to store data temporarily in the network. In the proposed scheme, two optimal storage zones Z1 and Z2 are formed. These storage zones contain a large number of storage nodes. Our goal is to prolong the network life time by finding optimal storage locations for data so as to minimize the data/query traversal path and thus to minimize total energy cost in data accumulation, transmission and data querying. To do this, scheme finds the sensing rate, query rate and hop count of the network. Based on these factors, it finds the optimal storage location using the DV-HOP and PSO algorithm for sensor zones so as to reduce the overall communication thereby reducing overall energy consumption.

II. Related Work

Data storage has become an important issue in sensor networks as a large amount of collected data need to be archived for future information retrieval. Data storage in WSN is addressed in either tree structure or mesh network topologies. Tree structures feature only one consumer (base station at the root) and multiple producers and do not take geographical location information into account when determining data storage placement. Whereas, a mesh network involves multiple producers and consumers yet the approach in the design of data storage to minimize communication overhead has been given to emphasize geographical locations with little attention to the data rates. Existing work on the problem of data storage follow two basic approaches. One is data-centric routing [8], and the other is data-centric storage [9]. In directed diffusion [8], queries are flooded in the sensor networks, and the data which are interested in are sent back to the consumer following the enhanced routes. This approach will be effective only when the queries are infrequent and are for streaming data type. Rumor routing [10] is another data-centric routing based data storage and retrieval algorithm. In rumor routing, when a node witnesses an event, it adds it to its event table, with a distance of zero to the event, and it also has a random chance to generating an agent. The agent is a long-lived packet which travels the network in a random direction, propagates information about local events to distant nodes and synchronizes the event table with every node it visits. Any node generate a query will forward the query in a random direction.

LEACH [11] is a clustering based routing protocol, in which cluster heads can fuse the data collected from its neighbors to reduce communication cost to the sink. LEACH has a similar structure to our scheme, having cluster heads aggregate and forward data to the sink. However, LEACH aims to reduce data transmission by aggregating data; it does not address storage problem in sensor networks. In [12][13] the authors propose a data-centric storage scheme for sensor networks, which inherits ideas from distributed hash table. The home site of a data is obtained by applying a hash function on the data type. Thus, queries for the same type of data can be satisfied by contacting a small number of nodes. To facilitate data query, Ganesan et al. [14] propose a multi-resolution data storage system, DIMENSIONS, where data are stored in a degrading lossy model i.e. fresh data are stored completely while long-term data are stored lossy. Ganesan et al proposed PRESTO[15] i.e. Predictive Storage in their work on storage architecture for sensor networks. A proxy tier is introduced between SNs and user terminals and proxy nodes can cache previous query responses. Compared to the storage nodes in this paper, proxy nodes in PRESTO have no resource constraints in term of power, computation, storage and communication. It is a more general storage architecture that does not take the characteristics of data generation or query into consideration. An energy-conserving data placement scheme proposed for sensor networks is introduced in [7] [16]. The authors propose a greedy heuristic that places multiple copies of data in the network and transfers the data from sensors to observers using multicast. Essentially, storage locations are the medians concerning communication costs among the senders and observers. In [17], a data storage placement scheme is proposed for a tree-structured sensor network where data are eventually gathered at the sink. Storage nodes are placed between the sink and the sensors to reduce energy consumption for data transmission. However, it collects these statistics only periodically and uses a greedy algorithm to compute the optimal storage position. Such a try-and-test greedy algorithm is computationally complex and makes it infeasible for a large-scale network deployments. Sheng et al. [16] utilized data rates, query rates, and compression ratio to determine storage placement and introduced storage nodes to alleviate the heavy load of transmitting all data to a sink/base station. They propose the optimal placement of multiple storage nodes but it can be only applied in a tree topology.

Zhaochun et al. [18] proposed optimal data storage (ODS) algorithms that can produce global optimal data storage position in linear, grid, and mesh network topologies. They formalized the data storage problem into a one-to-one (one producer and one consumer) model and a many-to many (m producers and n consumers) model with the goal of minimizing the total energy cost. They presented a near-optimal data storage (NDS) algorithm, which is an approximation algorithm and can obtain a local optimal position. Both ODS and NDS are locality-aware and are able to adjust the storage position adaptively to minimize energy consumption. But the main limitation of this work is the utilization of ODS and near-optimal solution. Lingzhi Zhu et al. [19] proposed a multi-optimal nodes data storage scheme (MODS), where multiple optimal nodes are chosen as the storage nodes. This scheme involves that producers send the acquired data to some storage nodes, which serve as data caches. Consumers get required data by sending queries to these storage nodes. Ranganathan et al [20] proposed hybrid particle swarm optimization algorithm to find the suitable positions for storage nodes while the total energy cost of data transmission is minimized. Clustering-based distributed data storage is utilized to solve clustering problem using fuzzy-C-means algorithm.

The most significant issue in the data storage approach is to gain suitable positions for a limited number of storage nodes in all nodes to make energy efficient, thus extending the lifespan of all wireless sensor networks. Most of the above works are based finding suitable storage nodes in the network. Finding the optimal position for these nodes is a big issue. In this scheme, we propose to find adaptive optimal storage locations for the storage nodes. The proposed scheme Energy Efficient Adaptive Storage Node Placement for Data Dissemination (EEASNPDD) forms two storage zones Z1 and Z2 in the network. These storage zones contain a large number of storage nodes. DV-HOP and PSO algorithm are used to find optimal positions for *k* storage nodes in WSN based on the energy cost of data transmission. Our goal is to prolong the network life time by finding optimal storage locations for data so as to minimize the data/query traversal path and thus to minimize total energy cost in data accumulation, transmission and data querying.

III. System Model and Assumptions

We assume a large-scale WSN comprising of homogeneous nodes. Network is represented as a directed graph G = (V, E), where *V* is the set of all nodes {S1, S2, …, Sn}and *E* is the set of edges between nodes that can directly communicate with each other in single hop i.e. are within communication range of each other. If node *Ni* can communicate directly with node *Nj*, a corresponding edge e_{ij} exists in E. The cardinality of *V* represents the total number of nodes N_t in the network, i.e. N_t=|V|. Without any loss of generality, each node in V is assigned a unique identifier. SNs are organized into clusters. Each cluster has a head node which functions as a server for all other nodes (i.e. clients) in the cluster. The head node maintains the links to the neighboring clients within its cluster and the heads in its neighboring clusters. It periodically exchanges “hello” messages with its clients and neighboring CHs. To communicate with its neighboring heads it specifies the IDs of the neighboring clusters in the messages. Also, we assume that the sink/base station has the topology information of the network. To formulate the data storage problem we have following assumptions.

(i). SNs are static.

(ii). SNs have similar capabilities for sensing, processing and communication.

(iii). A periodic data gathering application where data is sensed and is sensed and transmitted by each sensor to its cluster head (CH) and from the CH to another CH

(iv). All SNs have unique ID.

(v). All nodes have a common, maximum radio range equal to *d*. Thus, any pair of nodes, say *(I,j)*, can communicate if they are within distance *d* from each other i.e $dist(i, j) < d$.

(vi). To deliver their data, sensors may use multi-hop communications. Each SN independently sends data along the minimum-energy shortest path to the sink/storage zone. To compute the shortest path, standard Dijkstra algorithm is applied, assigning an energy cost to each link connecting two nodes that are within *d* distance from each other. The cost *C(i,j)* represents the total energy required to transfer a packet from node *i* to node *j* and is expressed by the sum of the cost at the transmitter and the cost at the receiver.

$$C(i,j)=E_t + E_r$$

(vii). The base station has the topology information of the network by monitoring and collecting the network. All nodes are aware of its location.

(viii). An event occurs randomly at any place at any time. A node *i* sends event data to or collects data from a storage node *k* at the fixed rate *R(i)* in a unit time interval

A standard flow problem in WSN includes two types of constraints, namely the flow conservation constraint (Data Flow Constraint) and the energy constraint.

$$\sum_{j\epsilon N_i} x_{ij}(t) = \sum_{j\epsilon N_i} x_{ji}(t) + y_i(t) \qquad (1)$$
$$\forall_i \in N, \forall_j \in N_i, \forall_t \in T$$

$$\sum_{t\epsilon T} \sum_{j\epsilon N_i} x_{ij}(T) * e_{ij} \le E_i \qquad (2)$$
$$\forall_i \in N, \forall_j \in N_i$$

The flow conservation constraint, equation 1, shows that the total amount of flow that a sensor receives plus the amount of data that it generates is equal to the amount of information that it transmits. where *t* (respectively *T*) is a time instance (respectively the network lifetime), *N* the set of sensors, N_i the set of neighboring nodes of *i*, x_{ij} the flow over the edge *ij* (data transmitted over this link), y_i the data generated by node *i*, e_{ij} the energy consumed in transmitting a unit flow and E_i the initial energy of the sensor. The second constraint given in equation (2) is the capacity constraint, which is related to energy. This constraint implies that the energy consumed by a sensor for transmitting the flow throughout the lifetime of the network must be less than its initial energy. The flow is represented by the number of packets and the transmission energy is calculated based on the distance between the nodes .The optimal solution of this problem gives an upper bound for network lifetime.

IV. Data/Query Path Setup

In sensor networks, SNs sense the event and send the sensed data towards sink node. Sink nodes issues the query and query travels multiple hops to get the desired result. On detecting an event for the first time, active SNs sense event and send readings to their cluster head (CH).To reduce the transmission of readings towards sink nodes; data aggregation is performed at CH. During a SN sensing interval (SNSI), CH aggregates similar readings from each active SN's in its cluster to yield a single data item. CH aggregates these readings to generate initial data item D_{in} and forwards it to its upstream CH's normally towards sink. But in our scheme, we propose to setup two storage zones Z1 and Z2 .SN send aggregated data D_{in} towards storage zones instead of sink node. Storage zones stores data that they receive from SNs. In addition these storage zones also accept the queries from the sink node. Each storage zone contains an Index Zone. This zone keeps record of the data stored in that zone. Each zone sends messages to source and sink about the kind of data stored in that zone. Low power radio mode of a SN is exploited to form a storage zones. In our approach, as shown in Figure 1, a circular region of radius $R_{L/2}$ around each CH is defined as storage zones. Let SZ is the set of nodes in storage zones. Nodes in *SZ* cooperate among themselves and with CH by sharing their local caches to realize much larger storage zone. All nodes in *SZ* communicate with each other and with CH using low power radio for every storage activity. The reason for taking radius of storage zones as $R_{L/2}$ is two-fold; first all nodes in storage zones should communicate in single hop with each other and second to utilize low power radio for caching so as to conserve energy.

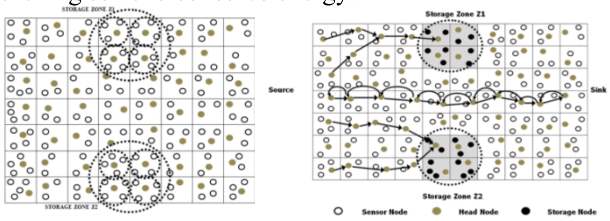

Fig. 1. Storage zones

The CH maintains the links to the neighboring clients within its zone, and the CH's in its neighboring zones. It periodically exchanges "*hello*" messages with its clients and neighboring zone heads. A CH communicates with its neighboring heads by specifying the IDs of the neighboring zones in the messages.

To further reduce energy consumption, in each zone, all nodes except the CH are in sleep mode. Nodes in sleep mode still can sense data, but rely on the head for other functions. The nodes rotate the responsibility of acting as the CH in a round-robin manner in the order of their identifiers to balance the workload and energy consumption among nodes. Each source node generates/samples data at a rate of r_s (times per unit time) and the size of each source data item is r_d. Sink issues queries for the collected data at a query rate q_i (times per unit time) to storage zones Z1 and Z2.SN's after sensing event keep on sending sensed data towards storage zones called push operation. Storage zones will store that data.

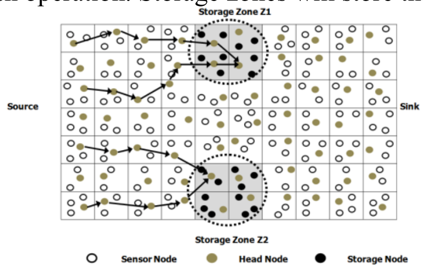

Fig. 2. PUSH Operation

Secondly, *Pull* operation is performed on stored data at storage zone. Sink node send queries to storage zones and storage zones will search that data and send the result to the sink nodes.

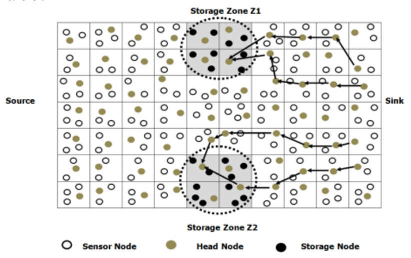

Fig. 3. PULL operation

The various symbols used are as follows

Table 1

Symbol	Definition
SN	Sensor Node
CH	CH is a SN that acts as cluster head.
SNSI	Sensor Node Sensing Interval is a period after which active SNs sense event
r_s	Source Node data rate
r_k	Sink Node data rate
q_i	Query Rate
r_d	Data size of source data
$C_{i,j}$	Communication cost from node I to node j
R(I,j)	Overall query rate
S(I,j)	Overall rate of data generation

We assume that total energy consumption consists of two parts: energy cost during push operation (Data Push Cost), energy cost during pull operation (Data Pull Cost). Let $C_{i,j}$ be communication cost of transmitting one unit of data over communication channel from node *i* to node *j*. The optimal storage node problem can be illustrated by sample graph shown in figure 4. Here, nodes 1 and 2 are storage nodes, nodes 3, 4 are source nodes and 0 is the sink node.

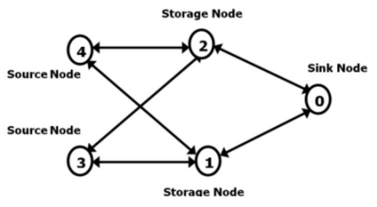

Fig. 4. Basic Scenario

Push Cost

Energy cost during *Push* operation is related to data size, distance between nodes and data rate. Cost of transmitting source data from source node 4 to storage node 2 per unit time is

$$r_s * C_{4,2}$$

Cost of transmitting source data from source node 3 to storage node 1 per unit time is

$$r_s * C_{3,1}$$

Total Push cost to from SNs to storage nodes is

$$C_{push} = r_s * C_{4,2} + r_s * C_{3,1} \qquad (3)$$

Pull Cost

Pull cost consists of cost of query and reply to that query cost.

$$C_{pull} = C_{query}+C_{reply}$$

Cost of query and reply to that query from storage zones transmitting storage data from storage node 2 to sink node 0 per unit time is

$$r_k * C_{0,2}+ r_k * C_{2,0}$$

Cost of transmitting storage data from storage node 1 to sink node 0 per unit time is

$$r_k * C_{0,1}+r_k * C_{1,0}$$

Total *Pull* cost from storage node to sink is

$$C_{pull} = 2*(r_k * C_{2,0} + r_k * C_{1,0}) \qquad (4)$$

Total Cost= Push Cost + Pull Cost

$$C_{total}(k) = C_{push}(i,k) + C_{pull}(j,k)$$

$$C_{total}(k)= \sum_{i=1}^{m} \sum_{j=1}^{k} r_s . C_{i,\, j} + 2 \sum_{q=1}^{l} \sum_{p=1}^{m} r_k . C_{q,p} \qquad (5)$$

Our goal is to find the set of storage nodes in *G* that minimizes overall communication cost i.e to determine the most energy-efficient storage position, the position where the energy consumption associated with data storage, transmission and query diffusion is minimal. Finding the optimal storage nodes i.e. to get the minimal in $C_{total}(k)$ above equation, is a challenging problem because many factors can impact the storage position selection, e.g. data rates, distance between SN's, network size etc.

V. Proposed Energy Efficient Adaptive Storage Node Placement for Data Dissemination (EEASNPDD) Scheme

In this section, we propose the Energy Efficient Adaptive Storage Node Placement for Data Dissemination (EEASNPDD) scheme to determine the optimal storage nodes k and storage zones. The purpose of EEASNPDD is to find an optimal set of nodes to store the data and respond to the queries.The dissemination model is of continuous monitoring type where SNs sense environment continuously over a long period of time. Each SN keeps on sensing the environment after every small period defined by Sensor-Node-Sensing-Interval (SNSI) and stores the values in its small buffer SN_{BUFFER}. After expiry of another interval called as Sensor-Node-Dissemination-Interval (SNDI) which is much larger than SNSI i.e. SN disseminates the entire buffer to its cluster head (CH). CH performs data aggregation and moves the data towards CH_{BUFFER}.CH_{BUFFER} disseminates it to sink node after every cluster-head-dissemination-interval (CHDI).Sink nodes issues queries towards SN's after regular intervals at a certain query rate (QR)

Sensing rate (SR) =1/SNSI

The storage of data in sensor networks is dependent on two main factors: First is data rate i.e. data sensing rate of source nodes and query rate. Sensing rate of source nodes is the rate at which SNs sense the event, generate sensed data and disseminate it towards storage nodes/sink node. Query rate is the rate at which sink node issues queries towards storage nodes/source nodes. The second is the path distance to the storage nodes. These two factors impact data storage-related communication cost in sensor networks. If the data rate is more than query rate, which means more and more sensed data is generated. The storage zones should be near to the source nodes as this will reduce the communication cost between sensed data and storage zones. If the query rate is more than data rate, then storage zones should be near the sink nodes thereby reducing the overall communication. If the data rate and query rate are same, then storage zones should be at the center of the network. Closer the storage zones to the source and sink, shorter the hop distance, the cheaper it is to store and query a fixed quantity of data. The optimal strategy is to place storage nodes adaptively according to data and query rate so that the communication cost is minimal. The find optimal location for storage nodes, we use fusion of Distance Vector by Hop counting (DV-HOP) algorithm and Particle Swarm Optimization (PSO) algorithm. If the SR is more than QR, then source node is set as anchor node and if QR is more than SR, then sink node is set as anchor node. DV-HOP algorithm is used to find the optimal location for storage node and number of hops. To increase the accuracy of the optimal location, PSO is fused with DV-HOP. After finding the optimal location for storage node using the algorithm, the nodes that are in communication range of storage node, form a storage zone Z1.

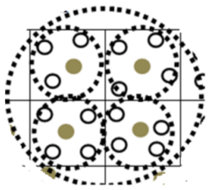
Fig. 5. Storage Zone

Storage zone Z2 is formed by finding the next optimal storage node that are not in the communication range of storage zone Z1.After forming the two optimal storage zones Z1 and Z2, the algorithm compares the SR and QR. If SR is more than QR, then storage zones are set as sink node and data from source node is sent towards storage zones. If QR is more than SR, then storage zones act as source nodes and process the query from the sink node.

VI. DV HOP ALGORITHM

The Distance Vector by Hop counting (DV-Hop) is the most known distributed algorithm. The DV-Hop algorithm was first reported by Dragos Niculescu and Badri Nath in Niculescu and Nath [21]. The algorithm estimates the distance between anchor nodes and unknown nodes by multiplying the hop count by average distance per hop, and then uses the three edge measurement method to estimate the coordinates of unknown nodes. The main idea of this algorithm is to determine approximate distance between two nodes by multiplying average hop distance with number of hops between them. It consists of three phases. In first phase, each anchor node sends it's *ID*, coordinates and hop count value (HopCount) (initially set to 0) in the form of packet *($id_i, x_i, y_i, HopCount_i$)* to its neighbor nodes. The neighbor nodes record the identification number of each node, the coordinate values and the smaller hop values. The packet is forwarded after hop values plus 1.If this value is less than the received one, then the latter is ignored; otherwise the receiving sensor increments the value of the received *HopCount*, updates its stored *HopCount*, then floods it in the network. In this way, all the nodes in the network get the minimum value of hop count from each anchor and location information of every anchor in the form of hop count table.The second phase calculate average hop distance between the unknown nodes and anchor nodes

$$\text{HopSize}_i = \frac{\sum_{i\neq j}\sqrt{(x_i-x_j)^2-(y_i-y_j)^2}}{\sum_{i\neq j} HopCount_{ij}} \qquad (6)$$

Where, (x_i,y_i) and (x_j,y_j) are the coordinates of anchor nodes i and j, and HopSize_i is the minimum number of hops between i and j. After computing the average distance per hop, the anchor nodes will transmit the information with TTL (time to live) in packet to the network. The unknown nodes will only record the first average distance per hop which it receives and transfer it to neighbour nodes. This strategy ensures that most of nodes receive the average distance per hop from the nearest anchor node. The estimated distance between the unknown node and anchor node is:

$$L_i = S_i * \text{HopSize}_i \qquad (7)$$

After getting distance from each anchor, in the third phase, the unknown node determines its location using multilateration method. This multilateration method uses least-squares technique. According to the distance between the unknown nodes and each anchor node, use the multilateral method to calculate the coordinates of unknown nodes, as shown in below equation:

$$\begin{aligned}
&(x_i-x_1)^2+(y_i-y_1)^2 = L_1^2\\
&(x_i-x_2)^2+(y_i-y_2)^2 = L_2^2\\
&(x_i-x_3)^2+(y_i-y_3)^2 = L_3^2 \qquad (8)\\
&\dots\\
&\dots\\
&(x_i-x_j)^2+(y_i-y_j)^2 = L_j^2
\end{aligned}$$

Where, (x_i,y_i) is the coordinate of unknown node; $(x_1,y_2),\dots,(x_j,y_j)$ are the coordinates of anchor nodes recorded by the unknown node. Subtracting the last equation from previous n-1 equations, simplifying and writing in matrix form, we obtain

$$AX=B$$

The estimation error of the least square method in traditional Distance Vector-Hop (DV-Hop) algorithm is too large and the Particle Swarm Optimization (PSO) algorithm is easy to trap into local optimum. In order to overcome the problems, a fusion algorithm of particle swarm algorithm and DV-Hop algorithm is presented. The node localization result are optimized by using the PSO algorithm in the third stage of the DV-Hop algorithm.

DV-Hop algorithm can be divided into three stages, in the first and second stages of DV-Hop algorithm, the distance $L_1,L_2,\dots,L_j$ between the unknown node $o(x,y)$ and the anchor node $A_1(x_1,y_1), A_2(x_2,y_2),\dots,A_j(x_j,y_j)$ is obtained by the hop count and the average hop distance between nodes, the ranging error is $\epsilon_1,\epsilon_2,\epsilon_3,\dots\dots\epsilon_j$ the estimated coordinates (x,y) satisfies the following inequalities:

$$\begin{aligned}
L_1^2+\epsilon_1^2 \le (x_i-x_1)^2+(y_i-y_1)^2 \le L_1^2+\epsilon_1^2\\
L_2^2+\epsilon_2^2 \le (x_i-x_2)^2+(y_i-y_2)^2 \le L_2^2+\epsilon_2^2\\
L_3^2+\epsilon_3^2 \le (x_i-x_3)^2+(y_i-y_3)^2 \le L_3^2+\epsilon_3^2\\
\dots\\
\dots\\
L_j^2+\epsilon_j^2 \le (x_i-x_j)^2+(y_i-y_j)^2 \le L_j^2+\epsilon_j^2
\end{aligned}$$

Positioning problem is transformed into finding coordinates (x,y) which minimize $f(x,y)$ of formula 11, and minimum $f(x,y)$ guarantees minimum total error. Therefore, the third stage of DV-Hop is converted into solving constrained optimization problem, finding $f(x,y)$is a nonlinear optimization problem .

$$F(x,y) =\sum_{i,j}^{n,m} |\sqrt{(x_i - x_j)^2 - (y_i - y_j)^2} - d_j|$$

Where m is the number of anchor nodes. Fitness function is used to evaluate the merits of the particle position and guide the direction of the search algorithm, the calculation is as follows:

$$Fitness_i = 1/m\sum_{i,j}^{n,m} |\sqrt{(x_i - x_j)^2 - (y_i - y_j)^2} - d_j|$$

Where $fitness_i$ is the fitness value of particle i, (x_i,y_i) is the position coordinates of the particles i, (x_i,y_i) is the location coordinates of the anchor node j, d_j is the distance between unknown node to the anchor node j.

VII. Particle Swarm Optimization (PSO)

The particle swarm optimization is a population based optimization technique, introduced by Kennedy and Eberhert in 1995[22]. The model of this algorithm is based on the social behaviour of bird flocking. The PSO is a mathematical computation technique that optimizes a problem. It is done iteratively by trying to find a candidate solution while maintaining a specified quality. The PSO includes a population of the candidate solutions which primarily called as the particles. In PSO, each single solution is a “bird” in the search space. We call it “particle”. All of particles have fitness values that are evaluated by the fitness function to be optimized, and have velocities that direct the flying of the particles. The particles fly through the problem space by following the current optimum particles. PSO is initialized with a group of random particles (solutions) and then searches for optimal by updating generations. Each particle, which is a potential global optimum of the function f(x) over a given domain D, is looked as a point in the D-dimensional space and represented as xi = $(x_{k0}, x_{k1}…x_{kn-1})$. Fitness value of all particles is evaluated by the fitness function to be optimized. According to that value, the particle is updated to move towards the better area by the corresponding operators till the best point is found.

In every generation, each particle is updated by following two “best” values. The first one is the best solution (fitness) it has achieved so far. (The fitness value is also stored.) This value is called pbest. $Pi = (p_{i1}, p_{i2}… p_{id})$. At the same time, the global best, which is the position with the best fitness value of all particles, is also recorded as $P_g = (p_{g1}, p_{g2}… p_{gd})$. Velocity, the rate of the position change for the i-th particle is represented as $V_i = (v_{i1}, v_{i2}…v_{id})$.At each times step, the velocity of all particles is adjusted as a sum of its local best value, global best value and its present velocity, multiplied by the three constants *w, c1, c2* respectively shown in equation 9.The position of each particle is also modified by adding its velocity to the current position shown in equation 10. In PSO a particle (individual) is composed of:

Three vectors:

- The **x-vector** records the current position (location) of the particle in the search space,
- The **p-vector** records the location of the best solution found so far by the particle, and
- The **v-vector** contains a direction for which particle will travel in if undisturbed.

Two fitness values:

- The **x-fitness** records the fitness of the x-vector, and
- The **p-fitness** records the fitness of the p-vector

$X = <x_{k0},x_{k1},…,x_{kn-1}>$
$P = <p_{k0},p_{k1},…,p_{kn-1}>$
$V = <v_{k0},v_{k1},…,v_{kn-1}>$
x_fitness = ?
p_fitness = ?

Particles are agents that fly through the search space and record (and communicate) the best solution that they have discovered. Particle moves from one location in the search space to another by adding the v-vector to the x-vector to get another x-vector ($X_i = X_i + V_i$).Once the particle computes the new Xi it then evaluates its new location. If x-fitness is better than p-fitness, then $P_i = X_i$ and p-fitness = x-fitness.The v-vector is calculated before adding it to the x-vector as follows:

$$v_{id} = v_{id} + c1*rnd()*(p_{id}-x_{id}) + c2*rnd()*(p_{gd}-x_{id}); \quad (9)$$
$$x_{id} = x_{id} + v_{id}; \quad (10)$$

Where

c1,c2 are learning rate / acceleration constants governing the **cognition** and **social** components

Where **i** is location of the particle, **g** represents the location of the particle with the best of p-fitness

Where, p is personal best

Where **d** is the dimension.

v_{id}	Velocity of the ith particle
p_{id}	pBest position of the ith particle
p_{gd}	the gBest position of the particles
x_{id}	current position of the ith particle
c1 & c2	are acceleration constants
r()	random function in the range [0, 1]
w	Inertia weight

The PSO algorithm is as follows

```
1. Begin.
2. Generate random population of $N$ solutions (particles).
3. For each individual $i \in N$, calculate $fitness(i)$.
4. Initialize the value of the weight factor $\omega$.
5. For each particle, set $pBest$ as the best position of particle $i$.
6. If $fitness(i)$ is better than $pBest$, then $pBest(i) = fitness(i)$.
7. Else set $gBest$ as the best fitness of all particles.
8. End.
9. For each particle, do calculation of particle velocity according to (3).
10. Update particle position according to (4).
11. End.
12. Update the value of the weight factor $\omega$.
13. Check if termination = true.
14. End.
```

The flow chart of the scheme is as follows

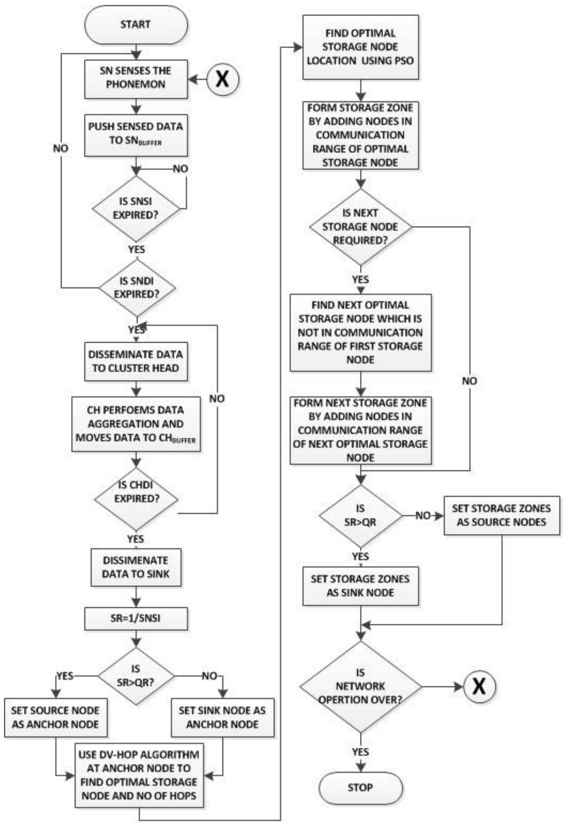
START
SN SENSES THE PHONEMON
PUSH SENSED DATA TO SNBUFFER
IS SNSI EXPIRED?
IS SNDI EXPIRED?
DISSEMINATE DATA TO CLUSTER HEAD
CH PERFOEMS DATA AGGREGATION AND MOVES DATA TO CHBUFFER
IS CHDI EXPIRED?
DISSIMENATE DATA TO SINK
SR=1/SNSI
IS SR>QR?
SET SOURCE NODE AS ANCHOR NODE
SET SINK NODE AS ANCHOR NODE
USE DV-HOP ALGORITHM AT ANCHOR NODE TO FIND OPTIMAL STORAGE NODE AND NO OF HOPS
FIND OPTIMAL STORAGE NODE LOCATION USING PSO
FORM STORAGE ZONE BY ADDING NODES IN COMMUNICATION RANGE OF OPTIMAL STORAGE NODE
IS NEXT STORAGE NODE REQUIRED?
FIND NEXT OPTIMAL STORAGE NODE WHICH IS NOT IN COMMUNICATION RANGE OF FIRST STORAGE NODE
FORM NEXT STORAGE ZONE BY ADDING NODES IN COMMUNICATION RANGE OF NEXT OPTIMAL STORAGE NODE
IS SR>QR
SET STORAGE ZONES AS SOURCE NODES
SET STORAGE ZONES AS SINK NODE
IS NETWORK OPERTION OVER?
STOP
YES
NO
X

Fig. 6. Flow Chart

VIII. Performance Evaluation

In this section, we evaluate the performance of proposed scheme energy-efficient Energy Efficient Adaptive Storage Node Placement for Data Dissemination algorithm (EEASNPDD) by performing simulations under different scenarios. The performance is evaluated by comparing it with ODS [12] and MODS[13] based on two performance metrics: *Total energy consumption* and Average Delay. Total energy consumption is defined as the overall communication energy consumed by the network to transmit and receive the data packets. Average delay is the delay in hops for a sink to retrieve the data from storage nodes. The default simulation setting has a square sensor field of size 400×400 m^2 in which N (400).SNs are uniformly distributed. Some of these SNs act as sources and generate data packet/messages. Simulation model is run 100 times and the observation is based on the varying numbers of SNs. There is one sink in the sensor field. The size of data packets are 40 bytes. The transmission range R of each sensor is 50 m. The focus of the evaluations is to study the network lifetime and residual power percentage of sensors by varying the number of SNs, data rate and storage node position. Table 2 summarizes various simulation parameters

Table 2. Simulation Parameters

Parameter	Value
Number of SNs	400
Size of sensing region	400X 400 m^2
Transmission Range	50 m
Data Packet Size	40 Bytes
Transmission Cost(per packet)	0.6 J
Receiving Cost(Per Packet)	0.4J
Distribution Type of SNs	Uniform
Type of Sensor Field	Two dimensional plane

8.1. Energy Consumption by Varying the Position of Storage Nodes

In this section, we evaluate the performance in terms of energy consumption with varying the position of storage nodes. The data rate of source means the data producing rate of SNs where as data rate of sink means data querying rate for sink node. The data rate of source and sink are considered same. Figure 7 shows the energy consumption with varying the position of storage nodes by placing them respectively at center, then near sink and finally near source nodes. We assumed that routing algorithm has established a path between source and sink. Figure 7 shows the total energy consumption comparison of different node placement strategies. From the figure 7, it is clear that energy consumption is least when the storage nodes are placed at the center as compared to placing storage nodes near source and sink. Placing the storage nodes at center of the field is the optimal storage location for storage nodes when date rates of source and sink are same.

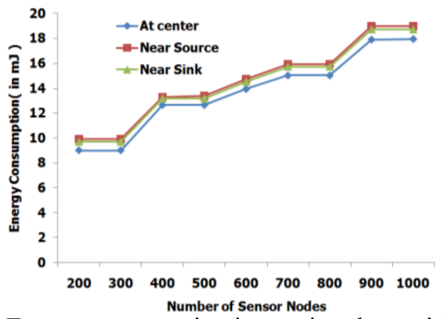

Fig. 7. Energy consumption in varying the position of storage nodes when data rate of source and query rate of sink are same

8.2. Energy Consumption of each Storage Node

In this section, we evaluate the performance in terms of energy consumption of each storage node with the number of SNs. The energy consumption of EEASNPDD is 15% less as compared to MODS and 35% less as compared to ODS. This is because in EEASNPDD storage zones are formed by group of SNs. This results in source node and sink node having a better chance to find nearest storage node and thereby reducing communication with storage nodes and thereby reducing the energy consumption. Figure 8 shows energy consumption by storage nodes in different strategies.

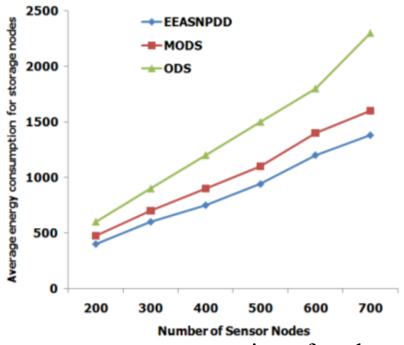

Fig. 8. Average energy consumption of each storage node

8.3. Average Energy Consumption of each Storage Node in Response to Queries

In this section, we evaluate the performance in terms of energy consumption of each storage node in response to queries from sink node. The energy consumption of EEASNPDD is 15% less as compared to MODS and 75% less as compared to ODS. This is because in EEASNPDD storage zones are formed by group of SNs. This results in sink node having a better chance to find nearest storage node and thereby query is served immediately and thereby reducing the energy consumption. Figure 9 shows energy consumption by storage nodes in different strategies.

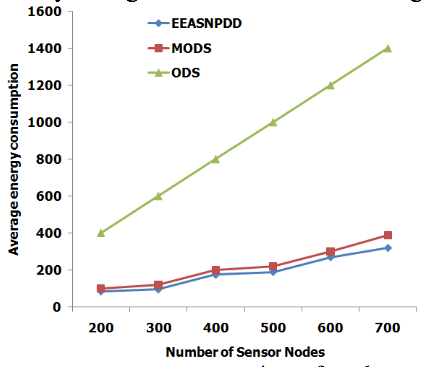

Fig. 9. Average energy consumption of each storage node in response to queries

8.4. Effect of Data Rate on Overall Energy Consumption

The change of data rate has huge impact on the energy saved using proposed EEASNPDD scheme. Proposed scheme focuses on setting the optimal storage location for storage nodes. Hence, this scheme gives maximum energy savings as compared to existing schemes. Figure 6 shows the total energy consumption comparison of EEASNPDD and ODS.

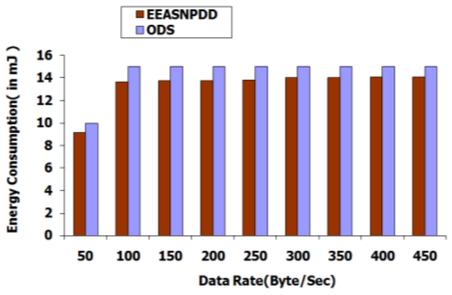

Fig. 10. Energy consumption in EEASNPDD and ODS when increasing the data rate

8.5. Effect of Number of Sensor Nodes on Average Delay

Average delay is the total number of routing hopes from sink to storage nodes to retrieve data. Sink node will issue query to the storage nodes and average delay is hops for a sink node to send query to nearest storage node. From the figure 11, it is clear that as the total number of nodes in network increases, the hop count decreases. This is because in our scheme of EEASNPDD, we have formulated storages zones that contain a number of storage nodes. This results in sink node having a better chance to find nearest storage node and thereby reducing average delay in the network. From the figure 11,it is clear that average delay in our proposed scheme is reduced by 12% as compared to MODS and 50% as compared to ODS.

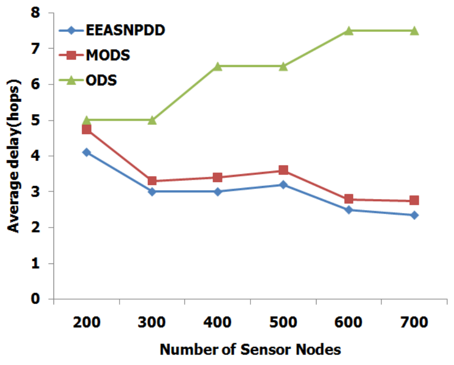

Fig. 11. Average delay in increased network size.

8.6. Overall Network Energy Consumption

In this section, we evaluate the performance in terms of energy consumption with time. Figure 9 shows the overall energy consumption with number of SNs. The overall energy consumption of EEASNPDD is 10% less as compared to ODS. This is because EEASNPDD places the storage zones are formed and number of storage nodes are available. Both source and sink have greater chance of finding storage node to nearest location thereby reducing the overall communication and thus reduces energy consumption. Figure 12 shows the total energy consumption comparison of different strategies.

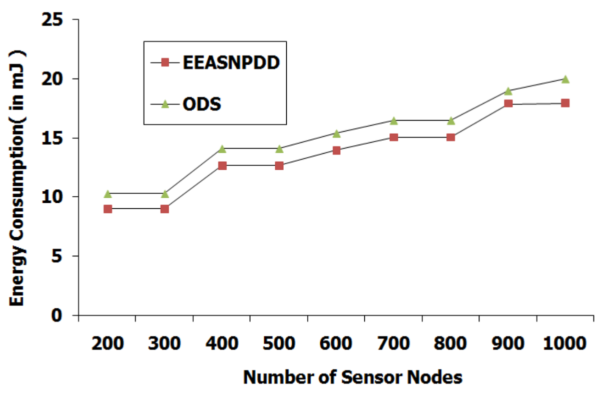

Fig. 12. Energy consumption of EEASNPDD and ODS in increasing the network size

IX. Conclusion and Future Work

This work attempts to find optimal storage locations for nodes in sensor networks and works out a push-push paradigm for data retrieval. To efficiently solve the storage problem, we propose a data storage strategy named Energy Efficient Adaptive Storage Node Placement for Data Dissemination (EEASNPDD). The proposed scheme is also useful for finding optimal storage nodes set for general topologies. Our goal is to prolong the network life time by finding optimal storage locations for data so as to minimize the data/query traversal path and thus to minimize total energy cost in data accumulation, transmission and data querying. To achieve this, we purpose to form two storage zones to store the sensed data and to process the query from sink. The scheme finds the optimal storage location using the DV-HOP and PSO algorithm for sensor zones based on data rate, query rate and number of hops. Simulation results show that EEASNPDD provides substantial energy saving as compared to existing schemes.

There are several directions for future research. First, our algorithm offers an optimal solution that requires the optimal placement of storage nodes. We plan to specifically incorporate the cost taken by storage nodes in determining the optimal storage set in our future work. Also, more sophisticated cost model that takes other factors, such as queuing effect and link quality, into consideration shall be further studied. Finally, it is a long-term goal to design an efficient protocol that acts dynamically to any topology change in a wireless network for an optimal set of storage nodes.

References

[1] I.F. Akyildiz, M.C. Vuran, O. Akan, W. Su, "Wireless Sensor Networks: A Survey Revisited," *Computer Networks Journal (Elsevier Science)*, vol. 45, no. 3, 2004

[2] K. Akkaya, M. Younis, "A Survey on Routing Protocols for Wireless Sensor Networks," Elsevier *Ad Hoc Network Journal*, vol. 3, 2005, pp. 325- 349.

[3] I.F. Akyildiz, M.C. Vuran, O. Akan, W. Su, "Wireless Sensor Networks: A Survey Revisited," *Computer Networks Journal (Elsevier Science)*, vol. 45, no. 3, 2004

[4] Ge-Ming Chiu, Li-Hsing Yen and Tai-Lin Chin, "Optimal Storage Placement for Tree-Structured Networks with Heterogeneous Channel Costs," *IEEE Transactions on computers*, vol. 60, no. 10, October 2011, pp. 1431-1444.

[5] P. Gupta and P.R. Kumar, "The Capacity of Wireless Networks," IEEE Trans. Information Theory, vol. 46, no. 2, pp. 388-404, March 2000.

[6] E.J. Duarte-Melo and M. Liu, "Data-Gathering Wireless Sensor Networks: Organization and Capacity," Computer Networks (COMNET), vol. 43, no. 4, pp. 519-537, Nov. 2003.

[7] B. Sheng, Q. Li, W. Mao, "Optimize Storage Placement in Sensor Networks," IEEE Transactions on Mobile Computing, vol. 9, no. 10, May 2010, pp. 1437-1450.

[8] C. Intanagon wiwat, R. Govindanj, D. Estrin, Directed diffusion: a scalable and robust communication paradigm for sensor networks, in: Proceedings of the6th Annual Int'l Conference on Mobile Computing and Networking (Mobi-COM'00), 2000, pp. 56–67.

[9] S. Shenker, S. Ratnasamy, B. Karp, R. Govindan, D. Estrin, Data-centric storage in sensor nets, SIGCOMM Comput. Commun. 33 (1) (2003) 137–142.

[10] D. Braginsky, D. Estrin, Rumor routing algorithm for sensor networks, in:Proceedings of the 1st Workshop on Sensor Networks and Applications, 2002.

[11] W. Heinzelman, A. Chandrakasan, and H. Balakrishnan. Energy-efficient Communication Protocols for Wireless Microsensor Networks. In International Conference on System Sciences, Maui, HI, January 2000.

[12] [G S. Ratnasamy, B. Karp, S. Shenker, D. Estrin, R. Govindan, L. Yin, and F. Yu. Data-centric storage in sensornets with GHT, a geographic hash table. Mobile Networks and Applications, 8(4):427–442, 2003.

[13] S. Shenker, S. Ratnasamy, B. Karp, R. Govindan, and D. Estrin. Data-centric storage in sensornets. SIGCOMM Computer Communication Review, 33(1):137–142, 2003.

[14] D. Ganesan, B. Greenstein, D. Estrin, J. Heidemann, and R. Govindan, "Multiresolution storage and search in sensor networks," ACM Trans.Storage, vol. 1, no. 3, pp. 277–315, 2005.

[15] M. Li, D. Ganesan, and P. Shenoy, "PRESTO: Feedback-driven data management in sensor networks," in Proceedings of the 3rd USENIX Symposium on Networked Systems Design and Implementation (NSDI '06), San Jose, CA, USA, May 2006.

[16] B. Sheng, Q. Li, W. Mao, "Data storage placement in sensor networks," Proceedings of the Seventh ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc'06), pp. 344–355, 2006.

[17] S. Shenker, S. Ratnasamy, B. Karp, R. Govindan, D. Estrin, "Data-centric storage in sensornets," ACM SIGCOMM Computer Communications Review, vol. 33, issue1, 2003, pp. 137–142.

[18] Zhaochun Yu, Bin Xiao , Shuigeng Zhou "Achieving optimal data storage position in wireless sensor networks" Elsevier 2009

[19] Lingzhi Zhua, Weimin Gaoa, Jun Honga, Xiaohua Deng "Adaptive information brokerage of multiple storage nodes in optical sensor networks "Elsevier 2015

[20] Ranganathan Mohanasundaram and Pappampalayam Sanmugam Periasamy "Hybrid Swarm Intelligence Optimization Approach for Optimal Data Storage Position Identification in Wireless Sensor Networks" Hindawi Publishing Corporation Scientific World Journal Volume 2015

[21] Niculescu, D., Nath, B., 2001. Ad-hoc positioning system. In: IEEE on Global Telecommunications Conference, vol. 5, pp. 2926–2931.

[22] Fengrong Zhang "Positioning Research for Wireless Sensor Networks Based on PSO Algorithm" ELEKTRONIKA IR ELEKTROTECHNIKA, ISSN 1392-1215, VOL. 19, NO. 9, 2013

Authors' Profiles

Vipan Arora have done B. Tech from NIT Hamipur (H.P), India and M.Tech from DAVIET, Jalandhar. Presently he is pursuing PH.D from NIT Hamirpur (H.P.). He is presently working as Head Computer Engineering department at Government Polytechnic College for girls, Jalandhar, Punjab, India. He has authored more than 25 books. His areas of interest includes adhoc and sensor networks.

Dr. T.P. Sharma have done doctorate from IIT, Roorkee. He is presently working as Associate professor, NIT Hamirpur (H.P). He has authored more than 70 papers in renowned International Journals. He has guided 5 PHDs and number of M Tech students. His areas of interest includes clustering, adhoc and sensor networks. email id: teekparval@gmail.com