Chapter 9
Data Centric Routing, Interoperability and Fusion in WSN

9.1 Introduction

Wireless sensor networks (WSNs) are mostly used for sensing and gathering data from environment on a continuous basis. The gathered data may pertain to various events related to monitoring and controlling applications such as:

- Environmental conditions,
- Vital parameters of patients,
- Movement and migration of habitats.
- Pest attacks on crops,
- Radiation levels in nuclear plant,
- Deep sea navigation,
- Seismic activities,
- Electronic surveillance, etc.

Accurate detection of specific events and monitoring of the events for the intervals of occurrences could be critical to implementation of certain functionalities of smart devices (smart phone, house hold appliances, or IoTs [1]). Furthermore, in response to these events, the behaviors of "things" or the devices must change in expected ways. The actuation of a specific behavior in a device is made possible through two important preparatory stages related to gathering of actuation data:

1. Dissemination of queries to sensors.
2. Collection of observed data from sensor nodes.

There are also passive monitoring applications, where the query dissemination is unimportant. Data collected through passive monitoring generally are collated by human experts and the actuation process is set in motion on the basis of a decision support system used by the human experts. So, it makes sense to consider the above two activities as parts of overall data gathering process using WSNs. Many sensors may report similar observations when they are in close proximity. The flow of redundant data generates unnecessary network traffic and affects performance. A smart

R.K. Ghosh, *Wireless Networking and Mobile Data Management*,
DOI 10.1007/978-981-10-3941-6_9

approach would be to fuse the data in some way to reduce the network traffic, and yet not lose any significant observation. As far as query dissemination is concerned, one idea could be to spread a query from the data sinks assuming bi-directional links. However, random spreading of queries may lead to congestion in WSN network due to redundant traffic. On the other hand, unicasting queries to the targeted nodes is not possible due to absence of node addressing mechanism for WSNs.

In Chap. 6, the focus was on ZigBee, 6LoWPAN and ZigBee IP protocols for low power wireless communication. These network protocols define IPv6 based standards for internetworking large number of sensors and connecting them to the Internet. Yet most of the existing WSN deployments, do not support any addressing mechanism for individual nodes. Any discussion on WSNs and their importance in realization of IoTs and smart environments remains incomplete without an understanding of the operational details concerning data gathering by WSNs. Therefore, in this chapter the focus of our discussion will be on the following four interrelated issues concerning data collection by WSNs:

1. Architecture of WSN.
2. Routing in WSN.
3. Integrating IP and WSN.
4. Data fusion in WSN.

9.2 Characteristics of WSN

Two of the most important technical limitations that distinguish the operations of WSNs from other network of computing devices as explained earlier are:

1. Lack of IP based addressing mechanism.
2. Resource scarcity at nodes, particularly the stored energy.

A WSN is a data centric network unlike Mobile Ad hoc NETwork (MANET). It is not possible to send a query to a specific sensor node. However, a query can be disseminated to sensor nodes for reporting values of a attribute or a class of attributes of the observed data. It is also possible to restrict dissemination of queries to a selected bunch of sensor nodes in close proximity to a location, or a region. Energy is a major resource bottleneck for continuous operation of WSN. Generally, it is assumed that the power consumption is mainly due to the operations of the radio interface [2–4]. Therefore, excessive attention has been placed in developing energy aware communication protocols.

Power consumption due to sensing module of a sensor node is generally ignored. However, emissions of toxic gases are measured in units of voltage levels. For accuracy in measurements, voltage levels are allowed to settle down for a while. Consequently, the sensing modules may consume comparatively more energy than the radio interfaces depending on the type of sensors being deployed. For example, NO_2, or VOC (Volatile Organic Compounds) sensors may at times can account for about

2/3rd of total energy consumption [5]. The application developers, therefore, have to understand the specification of a sensor in order to develop efficient code for sensor specific power control mechanisms. Kim et al. [5] developed an automated Power Management System (PMS) on the top of RETOS [6] operating system for sensor nodes . Following similar approaches, it may be possible to build PMS for many other types of sensor nodes. However, the utility of PMS may become obsolete with newer approaches based on the concept of energy packetization generally used in smart grid [7, 8]. The approach combines the transfer of power units with the communication from one source to another. So, the communication can still be carried out even when the intermediate nodes are about to die out. Novel energy aware routing schemes based on energy distribution through packetization can be developed in future for transmission of sensory data.

In summary, the following are some of the important characteristics of WSN which we need to keep in mind while dealing with WSNs:

1. Highly distributed architecture of WSNs.
2. Nodes in a WSN operate autonomously with very little local coordination.
3. Nodes in a WSN do not have any global addressing mechanism like IP, the communication is carried out in data centric mode.
4. Energy consumption is a major design consideration in operation of sensor nodes. So, communication and data processing must be performed in energy efficient manner.
5. Scalability of WSN depends on the density of the nodes per unit area.

9.2.1 WSN Versus MANET

An ad hoc network like a WSN is also a self organized wireless network of autonomous nodes which depend on battery. But the similarity between the two ends there. Sensor nodes are mostly static while the nodes in an ad hoc network are mobile. An ad hoc network is address centric and each node has a global IP address. Ad hoc nodes are represented by PDAs and handheld devices carried by the users who happen to be human. In contrast, sensor nodes are not carried by people. These nodes mostly are deployed in different environments some of which could be inaccessible. The number of nodes in a sensor network is several orders higher in magnitude than the number of nodes in a mobile ad hoc network. The sensors collect data objects of all kinds, related to both animate and inanimate things belonging to an environment. A node in ad hoc network in more powerful has more resources compared any sensor node. The routing protocols for mobile ad hoc network function

Table 9.1 Ad hoc network versus sensor network

Property	Ad hoc network	Sensor network
Hardware	Ad hoc nodes use the state of the art processors, and have reasonably better resources	Sensor nodes are cheap and have lower computing and communication capabilities
Deployment	Deployment is sparse. Depends on the number of active users in the area	Dense unattended deployment with large number of nodes. Typically many orders higher in magnitude than ad hoc network
Topology	Highly dynamic	May vary due to the variation in wireless signals but mostly static
Mobility	Ad hoc nodes are mobile	Sensor nodes are usually static unless mounted on vehicles, or worn by living beings
Routing	Address centric, each node has an IP address. Operate at IP layer	Data centric, operate at MAC layer
Energy	Uses stored battery power, but battery can be changed or recharged	Nodes operate in tighter power budget. Being deployed in unattended inaccessible places, battery change or charging is usually impossible
QoS	QoS is driven by the requirements to minimize network traffic	QoS is driven by the requirements to minimize energy
Applications	Distributed computing	Monitoring and data gathering

at network layer and require significant amount of resources whereas the routing in WSN typically function at MAC layer. Table 9.1 summarizes the differences.

9.3 Architecture of WSN

Building of tiny devices such as a sensor node was possible due to advances in Micro Electro-Mechanical Systems (MEMS) [9]. By inter-networking hundreds or thousands of sensors, a WSN is created. To give a background, let us begin with architecture of a sensor node.

At a very basic level, a sensor node as shown by the block diagram in Fig. 9.1 consists of three components:

1. A battery or power source,
2. A processor board with CPU, memory and sensing hardware,
3. A radio board with transceiver, which is connected by an antenna.

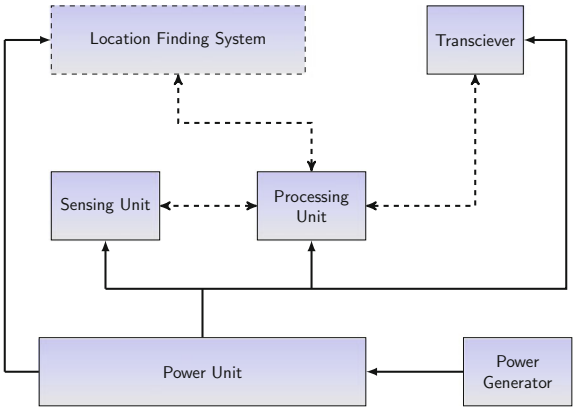

Fig. 9.1 Components of a sensor node

As the figure illustrates, the processor board connects CPU with sensing h/w and a small flash memory. There is a wireless transceiver which implements the physical and MAC layers, and connected to a radio antenna. Sensor nodes runs one of the many variants of OSes [10]. Among these, Tiny OS [11] and Contiki [12] are popular among the implementers.

9.3.1 Communication Architecture

The protocol level details and the communication architecture of sensor node were discussed earlier in Chap. 6 in context of ZigBee and 6LoWPAN. For the sake of completeness in discussion, we provide a logical view of communication architecture in Fig. 9.2. The network layer located in sensor board is responsible for routing, topology control, in-network data processing, data dissemination, storage and caching.

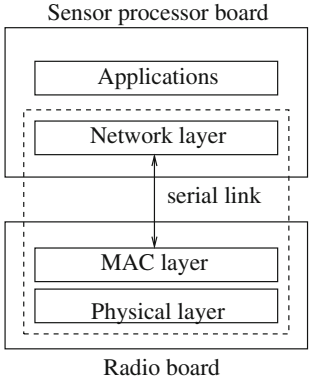

Fig. 9.2 Communication architecture of sensor node

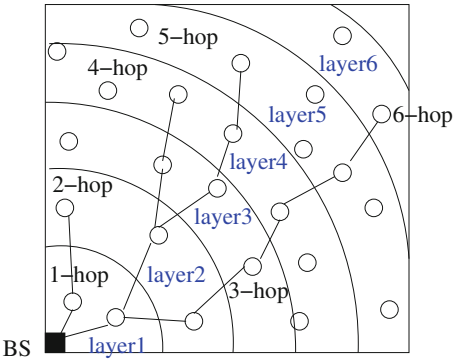

Fig. 9.3 Layered organization of sensor node

The MAC layer and the PHY layer are located on the radio board. The responsibilities of PHY layer include radio communication, access to sensing hardware, controlling actuation circuit, and signal processing hardware. The MAC layer is primarily responsible for sleep and wake up scheduling, and channel access schemes.

9.3.2 Network Organization

WSN use two basic type of organizations for data communication.

1. Layering,
2. Clustering.

Mostly the nodes are homogeneous and assumed to have a fixed communication range. Assume that sensor nodes are distributed in a rectangular region as shown in Fig. 9.3 with the base station (BS) being located at the left bottom corner of rectangle. In layering organization, the nodes 1-hop away from BS form layer-1, the nodes 2-hop away form layer-2 and so on as indicated in the diagram. The problem of funneling is quite common in a layered organization. It happens due to the fact that the nodes closer to BS die out must faster compared to the nodes at the periphery of network.

Clustered organization was proposed primarily to eliminate funneling problem. Figure 9.4 illustrates how clustered organization communicate data to the BS. Basically, each cluster head act as an intermediate data sink for the nearby nodes which are referred to as regular nodes. A network may have about 5–10% nodes as cluster heads depending on the requirements. Cluster heads may in turn form a layered organization. In practice, most of the cluster network topologies are two tiered, though it is also possible to create complex hierarchical topologies. Cluster heads are responsible for data aggregation of their respective clusters before data gets routed to BS.

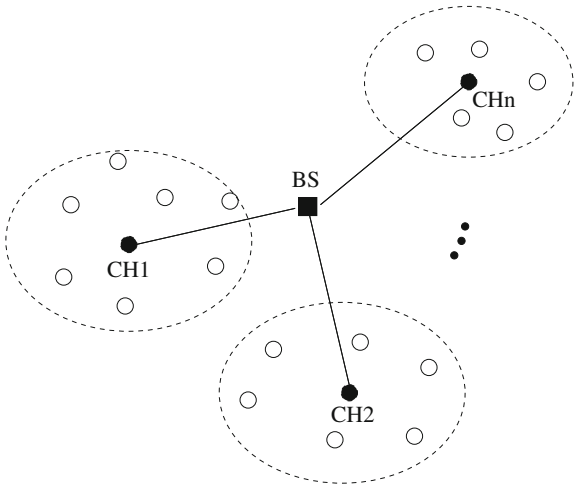

Fig. 9.4 Clustered organization of sensor node

9.4 Routing in Sensor Network

Let us first examine some of the basic characteristics of sensor networks which influence design of routing protocols. One of the important difference from IP networks is addressing of the sensor nodes. Typically, hundreds and thousands of sensor nodes form a WSN. The nodes have to establish connectivity irrespective of distribution of nodes in the network. Most applications do not care about the node IDs from which data is received but are interested only in observed data from certain locations.

Mostly the sensor nodes are static. As opposed to this in conventional IP networks, the nodes may have slow discrete movements. In most cases, a sensor network is deployed to serve only one specific application. Knowing geographic positions (latitude, longitude) of a node is advantageous in some applications as the observed data can be meaningfully associated with a specific geographical area. However, using GPS hardware in a node is infeasible due to following reasons:

- GPS does not work indoors, it may not even work in dense jungle, construction sites or deep canyons.
- GPS operation requires some power. Energy drain out occurs comparatively faster when GPS hardware is integrated with a sensor node.
- Integrating GPS with sensor node also increases the cost of sensor nodes.

Therefore, localization (determining position) of nodes is performed through various geometric and approximate method based on the idea of trilateration [13]. Trilateration methods are not only compute-intensive but also error prone. Observed data exhibit certain amount of locality. So, there may be a fair amount of redundancy in data flowing into base station. Eliminating redundancy in data flow could extend the life of the network. Considering the inherent characteristics of WSNs in mind, the design of routing protocols

- Should not depend on any global addressing schemes like IP.
- Should balance energy levels of the nodes.
- Should be able to handle sudden failures of the nodes, and unpredictable changes in topology.
- Should work without GPS or any precision positioning system.
- Should be power aware and minimize redundant information flow by applying various data fusion techniques.

Lack of global addressing mechanism is not a big problem, as a WSN is more a data centric than a node centric network. The information is requested based on the attribute values. For example, suppose we have a deployment of sensors to measure environmental temperature. Then the query may be of the form: "Is temperature is above 50 °C?" Then, only those nodes which have measured value $v > 50$ should report their measurements. The discovery and maintenance of routes in a WSN are nontrivial problems. This is due to the fact that WSNs have to operate with low energy budgets. Some nodes may die out quickly if network traffic is not uniformly shared by all the nodes. Even without GPS, locations can be prestored in the nodes before deployment, or locations can be estimated after the deployment by using trilateration techniques. So, the position information of the nodes can be used to route the data to the desired region rather than flooding into the whole network.

9.4.1 Classification of Routing Protocols

One of the excellent surveys on the routing protocols in sensor networks is made by Al-Karaki and Kamal [14]. There are several ways to classify the routing protocols. One classification scheme based on the network structure classifies the protocols into three classes, namely,

1. Flat-network based routing.
2. Hierarchic-network based routing.
3. Location-based routing.

In a flat network structure, every node is equal in status, and perform the same set of operations. In hierarchical network structure, the nodes have an implicit difference in roles and perform operations according to their assigned roles. In location based routing, the nodes exploit positional advantage to guide data to flow in and out of the selected region in the network. Some routing protocols can adapt to the changing network conditions such as topology change, change in energy level of nodes, etc.

Another way to classify routing protocols is based on the operation of protocols. It defines five different types of protocols, namely,

1. Negotiation based routing.
2. Mutilate based routing.
3. Query based routing.
4. QoS based routing.

5. Coherent based routing.

To give a flavor of various routing protocols, we plan to discuss only a few well known protocols. The readers who are interested to explore more, may refer to the original survey paper by Al-Karaki and Kamal [14].

9.5 Flat Network Based Routing

In this class of routing algorithm, every node has the same role, and they cooperate together to distribute data over the entire network. Typically, in such networks, the base station sends out queries for data either from a selected region or on the basis of certain attributes and waits for the same to arrive. Each sensor having data matching to a query responds. To facilitate the responses, each query should contain an accurate description of the needed data. A number of data centric routing protocols exist in literature. It is neither possible nor the focus of this book to discuss all the routing protocols. A short summary of SPIN [15] is provided as it is one of the earliest work in the area which focuses on energy saving issues.

SPIN (Sensor Protocol for Information via Negotiation) was proposed by Heinzelman et al. [15]. It belongs to a family of adaptive routing protocols. SPIN treats each node as a potential base station or data sink. So, any node may be queried to obtain all information about gathered data. Basically, the approach is to eliminate three common problems noticed in most sensor routing protocols:

- *Implosion*: The nodes in classical routing algorithms work by flooding, i.e., sending data to all the neighbors. It leads to the same data being received by a node multiple number of times as shown in Fig. 9.5(a). Node E receives data item d multiple number of times.
- *Overlap*: The data from the same overlap region arrives via different paths which results in increases in network traffic and leads to the wastage of bandwidth, and the energy. This problem is illustrated by Fig. 9.5(b). Node C receives overlapping data $d1$ and $d2$ multiple number of times.
- *Resource blindness*: No node cuts down its activities, even if the energy level is low. Since the nodes operate with pre-stored power resources, cutting down activities of the nodes with low energy level helps to extend the overall network lifetime.

SPIN is based on two main ideas, namely

1. Negotiation, and
2. Resource-adaptation.

SPIN uses negotiation before sending data. It uses principle of locality to control data dissemination. The underlying idea is that the nodes in same neighborhood most likely have similar data. So, each node should only distribute those data which other nodes do not have. A node on receiving new data broadcasts the meta data in the form of advertisements (ADV messages) to the neighborhood in the first stage. Meta

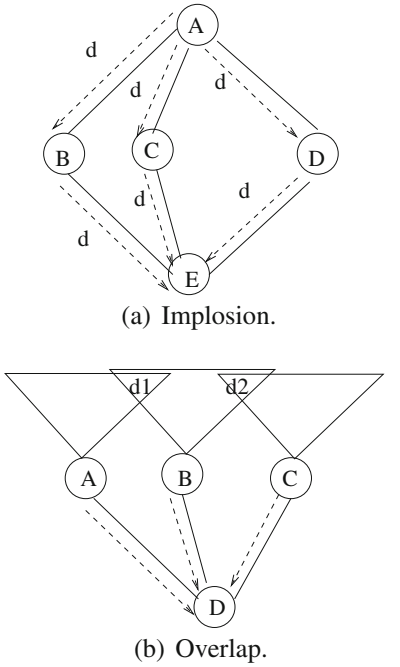

Fig. 9.5 Implosion and overlap problems

data is a short synopsis of data which the node possesses. SPIN does not specify the format of the meta data, the format is decided by the application developers. However, the guiding principles for the creation of meta data are:

1. The meta data descriptors of distinguishable data are also distinguishable.
2. Likewise the meta data descriptors of the non-distinguishable data are also non-distinguishable.

The nodes which receive ADV messages (meta data) check whether they have the corresponding data. If not, then they send out request (REQ messages) for the advertised data. On receiving REQ message, the advertising node sends DATA to the requesting node. The same three stage process {ADV, REQ and DATA} of data dissemination repeated at a new node. This way the data propagates throughout the whole network. Each node also has a resource manager to monitor its current level of resources. Before processing or transmitting data, the application needs to check resource level by querying the resource manager. Depending on energy threshold, the node may reduce its participation in the protocol. A node may participate if it can be sure of completing the three stage distribution process.

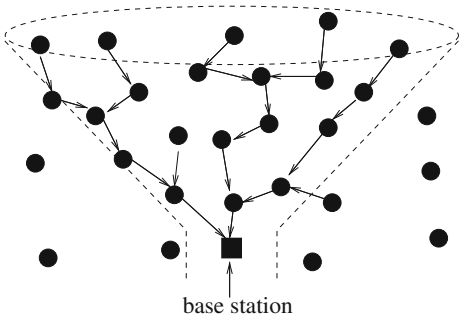

Fig. 9.6 Funneling effect in WSN

9.5.1 Hierarchical Routing Protocols

Hierarchical routing protocols defines two separate roles for the sensor nodes, viz., cluster heads and regular nodes. The idea of having certain nodes to perform the roles of cluster heads is mainly to eliminate funneling effect on nodes near the base station (BS). Figure 9.6 illustrates funneling effect. It occurs due to the fact that in multihop routings, energy drains out faster from the nodes near the BS than the nodes at the periphery. Since, the drain out of energy is not evenly distributed among the nodes, some of the nodes end up with shorter lifetime than the others. To mitigate the funneling effect the concept of 2-tier routing was proposed [16]. The role of cluster heads is assigned to nodes with higher energy levels. The nodes having lower energy levels, perform sensing and send the observations to a cluster head in proximity. A cluster head, typically, performs data aggregation/fusion. It reduces the number of messages needed to transmit the data pertaining to observations made by a set of sensors deployed in a region. Apart from mitigating funneling effect, the 2-tier routing protocols also contribute to the overall efficiency, the scalability and extending the lifetime of a network [14].

Low Energy Adaptive Cluster Hierarchy (LEACH) [16] is an example of two-tier routing algorithm. It uses a randomized rotation approach to assign the roles of cluster heads to the sensor nodes so that the battery of no particular sensor node drains out faster than the others. LEACH also applies compression of data by fusion technique to optimize the volume of data transmission from the cluster heads to the base station.

The cluster heads are elected from among the regular nodes. The number of cluster heads is determined in advance. It depends on a number of network parameters such as topology, relative costs of computation versus communications [16]. A node n chooses a random value v, where $0 < v < 1$. If $v < T(n)$, then the node becomes cluster head for the current round r. The threshold $T(n)$ is defined as follows:

$$T(n) = \begin{cases} \dfrac{P}{1 - P * (r \mod \frac{1}{P})}, \text{if } n \in G \\ 0, \text{otherwise} \end{cases}$$

where,

P is the desired percentage of cluster heads in the network, and
G is the set of nodes that have not been cluster heads in the last $\frac{1}{P}$ rounds.

For example, if 5% nodes are to be assigned the role of cluster heads then $P = 0.05$. Initially, $r = 0$, and each node has the probability P to become a cluster head. Once a node becomes a cluster head then it cannot become a cluster head for subsequent $\frac{1}{P} = 20$ rounds. The nodes which were cluster heads for round number r, become eligible for cluster head role in $r + \frac{1}{20}$ round.

The cluster heads broadcast an advertisement offering data aggregation service to the rest of the other nodes. The cluster heads use CSMA MAC protocol for the advertisement using the same transmit energy. The regular nodes should be in listen mode during this phase of the algorithm. Once the first phase is over, the regular nodes will know which cluster they belong to. This decision will be based on the RSSI of advertisement messages from the various cluster heads. The nodes then send their leader acceptance message to their respective cluster heads. Again CSMA MAC protocol is used for this transmission. During this phase the cluster heads must keep their receivers on.

Based on the received acceptance messages, each cluster head creates a TDMA schedule for receiving transmission from its cluster members. This schedule is broadcast back to the members. After the transmission schedule has been decided, transmission begins. Each node sends it data during its allocated transmission schedule. To receive the transmission the cluster head must keep its receiver on. After all transmissions are received, each cluster head applies compression and fusion techniques to create a consolidated message for the data received from the region under it. In the last phase of a round, a cluster head sends the report about the data pertaining to its region to the base station. Since the base station may be located far away from a cluster head, this is high energy transmission phase. The important conclusions from LEACH experimentation are as follows [16]:

1. LEACH cuts down the requirement for communication energy by a factor of 8 compared to direct transmission protocol.
2. The first death in network occurs 8× times later.
3. The last death in network occurs 3× times later.

9.5.2 Location Based Routing Protocols

Location of a node is the fundamental element of information used by this class of routing algorithms. The relative positions of the source and the destination provides a sense of direction in which the data should flow thereby restricting the flooding. The localization or the estimation of the positions of nodes is a one time task as the sensor nodes are mostly static. There are many GPS less localization algorithms that can estimate locations of the nodes by exchanging information between neighbors [13,

17–19]. Most of the algorithms in this category apart from restricting flooding also prescribe a sleep and wake up schedule for the nodes in order to save energy [20, 21]. Some of the location-based algorithms were developed for mobile ad hoc networks [20, 21], but are also applicable to sensor networks.

As an example, we discuss Geographical and Energy Aware Routing (GEAR) [22], which is motivated by the fact that most of the time the queries for data are location (region) related. In other words, a query packet is routed to a set of nodes (forming a region or a neighborhood) instead of a single node. It assumes every node is aware of its current location.

GEAR relies on recursive dissemination of a query, and considers energy level in nodes while selecting neighbor for forwarding the queries. There are two phases in the process to disseminate a query to a targeted region of a network, namely,

Phase 1: It uses energy aware neighbor selection to push the query to the target region.
Phase 2: Inside the target region, it uses two different rules to forward the query to the destination.

In the first phase while packets are pushed towards the target region, the residual levels of energy is considered.

- If there is a node closer to the target region than the query forwarding node, then the query is forwarded to all such nodes.
- Otherwise, the distance is also used in combination with the energy to get around an energy hole.

In the second phase, the query gets disseminated inside the target region. Either a recursive geographic forwarding, or a restrictive flooding is used. The recursive geographic forwarding works as follows. A node N_i receiving the query creates four sub regions. Then it creates four copies of the query and sends one copy to each of these four sub regions. The recursive query dissemination is carried out until the farthest point of the region is within the direct range of the forwarder but none of the neighbors are within the region. However, the recursive splitting may be expensive when the density of sensors is low. Additionally also it may create routing loops. So in this case pure geographical distance metric is used for efficiency.

The most important idea that can be termed as the main operating logic of the protocol is computation of the energy aware neighbor. The centroid of the target zone R is considered as the logical destination location D. On receiving a packet, a node N sends the packet progressively towards the target region while balancing the energy consumption across all the neighbors.

Each node N maintains a cost $lc(N, R)$, known as learned cost, from zone R. N infrequently updates $lc(N, R)$ to its neighbors. If a node does not have $lc(N_i, R)$ for a neighbor N_i, then it estimates a cost $ec(N_i, R)$, given below, as a default value for $lc(N_i, R)$.

$$ec(N_i, R) = \alpha d(N_i, R) + (1 - \alpha)e(N_i)$$

where,

1. α is an adjustable value in range (0, 1),
2. $d(N_i, R)$ is distance of N_i from centroid D of region R, and
3. $e(N_i)$ is the consumed energy at node N_i normalized by the maximum energy consumed across all the neighbors.

Once a node N has selected a neighbor N_s, it updates its own energy status as follows:

$$lc(N, R) = lc(N_s, R) + C(N, N_s),$$

where $C(N, N_s)$ is the energy consumed for sending a packet on link (N, N_s). $C(N, N_s)$ is a function of residual energy levels at N_s and N and the distance between N and N_s.

A discerning reader will notice that *ec* is a weighted sum of distance of a neighbor from the target region, and the normalized consumed energy at the neighbors. The estimated function, *ec*, is basically derived from an obvious considerations of the energy drain out principle. It is borne out from following two significant observations:

- In circumstance where all the neighbors have same level of residual energy, the value of *ec* depends on distance from the target region. Consequently, it amounts to selection of the neighbor using a greedy approach, where the node closer to the target region becomes the automatic choice.
- If, on the other hand, all the neighbors are at the same distance from the target region, the choice is dependent on the local minimization, which is essentially a load balancing strategy.

9.5.3 Selection of Forwarding Neighbor

Having understood the cost functions, let us find how selection of forwarding neighbor is carried out. Two possible scenarios that need to be examined are:

1. Each packet P carries ID of the target region R. If there are neighbors close to R, then a neighbor having minimum $lc(N_i, R)$ value will be picked up for forwarding. So, P will be routed progressively closer to R while balancing energy usage.
2. If all neighbors are far away from N itself, then N is said to be in a hole. When there is no hole the learned cost $lc(N_i, R)$ and the estimated cost $ec(N_i, R)$ are the same. If there is a hole, then the learned cost and the update rule help to avoid the holes in the path to destination.

To see how it works, we consider the example quoted in the original paper [22]. Figure 9.7 illustrates a grid deployment where three nodes marked G, H and I have no residual energy to participate in forwarding. The target node is T and packet originated from S. Assume that grid nodes are placed at equal distance from its east,

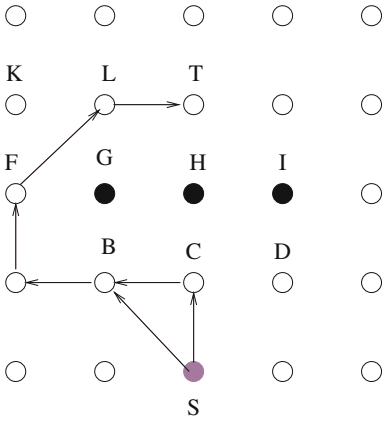

Fig. 9.7 Geographical forwarding in a grid deployment [22]

west, north and south neighbors. Without loss of generality, we may assume these distances to be one unit each. The grid layout imply each node has 8 neighbors: east, north-east, north, north-west, west, south-west, south, south-east.

Initially, at time 0, learned costs and estimated cost will be same.

$$lc(S,T) = 3, lc(B,T) = ec(B,T) = \sqrt{5}$$
$$lc(C,T) = ec(C,T) = 2, \quad lc(D,T) = ec(D,T) = \sqrt{5}$$

On receiving a packet P, S forwards it to the lowest cost neighbor, i.e., C. But C finds itself in a hole as all of its neighbors are a larger distance than itself. So, C forwards it to the neighbor N with minimum $lc(N,T)$. If ties occur, then they are broken by node ID. This implies among the neighbors C will pick B, and update its learned cost

$$lc(C,T) = lc(B,T) + C(C,B) = \sqrt{5} + 1$$

At time 2, when node S receives another packet P' for same region T, the values of learned cost for B, C and D are as follows:

$$lc(B,T) = \sqrt{5}, lc(C,T) = \sqrt{5} + 1, \text{ and } lc(D,T) = \sqrt{5}$$

So, S has a choice to forward P' to B or D, but B's ID is lower than D. Therefore, S will forward P' to B instead of C. In actual execution of the algorithm, packet forwarding from S will oscillate between C and B. Once a packet is delivered to T, the learned cost (LC) value will be propagated 1 hop back, and every time a packet is successfully delivered LC-value will be propagated 1 hop back, and finally it will converge after delivery of n packets if distance from S to T is n hops. Intuitively, LC-value together with update rule assist the scheme to avoid holes. For further details in this regard, the readers may refer to [22].

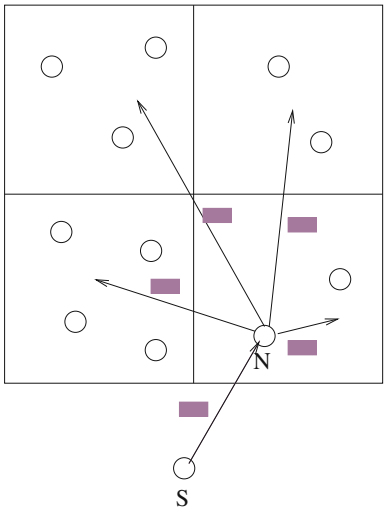

Fig. 9.8 Recursive geographical forwarding [22]

After a packet P reaches the target region R it can use simple flooding with duplicate suppression method to guide the packet to actual destination. A recursive forwarding method has been proposed in LEACH to control flood. Since, packets are forwarded to the centroid of target region R, the bounding box of R can be split into one subregions as illustrated in Fig. 9.8, and one copy of packet P is sent to the centroid of each of the subregion recursively. The recursive forwarding is continued until the stopping condition is met. The stopping criterion is met when either a region has no other node in its subregion. To determine the stopping condition we check if the farthest point of the region is within the range of the node but none of its neighbors are inside the region.

9.6 Routing Based on Protocol Operation

The set of routing protocols we will discuss in this section are based on characteristics of operation of protocols. A protocol may be of three types:

1. Multipath based routing.
2. Query based routing.
3. Negotiation based routing.

Multipath routing uses multiple paths to a destination. This category of routing protocols enhance the network performance and are resilience to failure of the links. In query based routing, source nodes propagate the query for fetching the data. The nodes having matching data send replies back to the node that initiated the query. In negotiation based protocols, the routes are obtained through negotiation based on the QoS metrics such as delay, energy or bandwidth, etc. The main advantage of a

negotiation based routing protocols is that it eliminates redundant transmissions of data which is specially important for a resource poor networks. Let us have a closer look at some instances of these three classes of routing protocols.

9.6.1 Multipath Routing Protocols

Since, multiple paths to a destination are used, these routing protocols have overhead in terms of generating higher network traffic compared to those which use single path. Alternative paths are maintained by sending control packets from time to time. These protocols, therefore, assure increased reliability in data delivery and resilience to failures. The resilience can be measured by the number of paths available between the source and the destination. Many instances of multipath routing protocols are available in literature [23–28].

Most of the protocols in this category, focus on the energy issues and choose to activate one of the available paths on the basis of either the residual energy on the path or pro-actively route data to maintain a balance of energy at different nodes. The idea is to extend the network lifetime as well as reliability. In [24], the focus is particularly on enhancing the reliability. It introduces a function for analyzing the trade-off between the degree of multipath and the failing probabilities of the available paths. It partitions a packet into subpackets with an added redundancy and sends each subpacket through one of the available paths. Due to availability of redundant information, the reconstruction of the original packet is possible even if some subpackets are lost.

The other interesting idea in class of multipath routing algorithms is presented by directed diffusion [26]. It uses the idea of publish and subscribe approach to send data of interest to the inquirers. An inquirer expresses an interest I. The sources that can service the interest I send the data to the inquirer. The inquirer effectively turns into a data sink and floods the interest. A forwarder typically maintains a local cache of the interests and stores an incoming interest if it is not already present in the cache. Initially an interest is diffused at a low data rate. Bidirectional gradient is established on all the links during the flooding. The sensor system samples data at the highest rate of all the gradients. The data is sent out using unicast as the reverse links are formed while the gradient is established. The initial low rate is known as exploratory. Once a sensor is able to report data, the reinforcement of the gradient occurs. Positive reinforcement is triggered by the data sink resending the interest with shorter intervals. The neighbor node witnessing the higher rate positively reinforce at least one neighbor that uses its data cache. It usually selects an empirically low latency path.

9.6.2 Query Based Routing Protocols

Directed diffusion [26] also represents a query based routing algorithm as interest diffusion essentially amounts to sending queries. Another instance of query based algorithm is presented by rumour routing protocol [29]. It essentially works in two stages. Initially, long-lived agents are used to create paths to the events when they occur. When a query related to the event is generated, it gets routed along that path.

The basic idea is as follows. The agents are basically long-lived messages circulating in the network. Each node maintains a list of its neighbors and a table of events with forwarding information to all events. The events may have an expiration time to restrict size of the event table. When a node generates an event, it probabilistically creates an agent with route length 0. The probability is used, because other nodes noticing the event, may also create agents. It may be too much of overhead if each node noticing the event creates a separate agent. The agent then travels in the network with a TTL (time to live) value in terms of the number of hops. When an agent travels, it combines the event table of the originating node with the visited nodes. It also updates the event table when a shorter path is found at a visited node. A node generates a query unless it learns a path to a required event. If no route is available, the node transmits the query to a random direction. If the response is not received after a finite wait time, the node then floods the network with the query. Clearly, the routing protocol has a significant overhead due to agent circulation, random query initiation, etc.

9.6.3 Negotiation Based Routing Protocols

Negotiation based routing are typically the choice for QoS requirements. However, as explained in SPIN protocol [15] earlier in the text, negotiation can be used for suppressing unnecessary transmissions. In fact, even in QoS requirement based routing, the path which cannot provide required QoS, will also not receive unnecessary network traffic. There are many other examples of routing protocols [30, 31] where negotiation is the main basis for routing.

In general, QoS metrics could be specified by

1. Application,
2. Protocol's overall goal,
3. Protocol's strategy to route data.

For example, at the application level QoS may refer to the frequency of data, and the quality of data. At protocol level, the strategy could be the shortest path, on occurrences of events, etc. The overall goal of a routing protocol could be extending lifetime, minimizing energy, etc. Most QoS based algorithm will require frequent recomputation of the routes as existing ones may no longer promise QoS after extended periods of use.

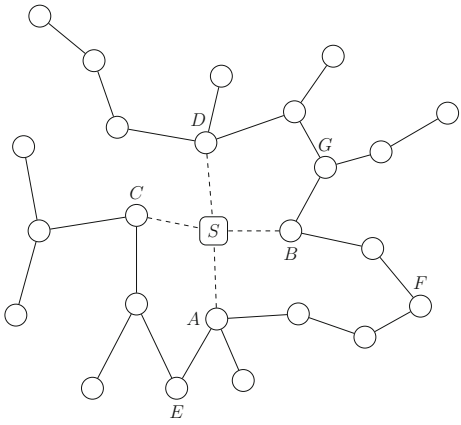

Fig. 9.9 Sequential assignment routing algorithm

Sequential Assignment Routing (SAR) [31] is stated to be one of the earliest routing algorithm that utilized negotiation [14] involving factors such as energy, QoS on path and the priority levels of the packets. SAR creates multiple trees with root of the tree and the data sink being within each others ranges. Each tree grows outward from the sink avoiding the nodes with low throughput or high delay. It can provide multiple paths for the data to be transmitted from each node to the data sink. So, the higher priority data packets can be transmitted via low delay paths, at the same time the lower priority packets from the same node can take a higher delay but a higher residual energy path. An example is provided in Fig. 9.9. As the figure shows, nodes E, F, G have alternative paths to sink. E can be reach S via A or C with paths of length 2 and 3 respectively. Similarly, F can reach S through paths of lengths 4 and 3 respectively from A and B. G can reach data sink S either through B or through D with respective path lengths of 3 and 2. Depending on which path provides better value for chosen QoS metric, the data to sink can take one of the possible alternative paths.

9.7 Interconnection of WSNs to the Internet

Initially, the objective of WSN research was on data gathering connected with specialized systems for automation, control and monitoring. Instead of standardized protocols, highly specialized protocols using one or more gateways were considered as adequate for accessing WSN data over the Internet. However, WSNs without IP integration offered an inflexible framework for the development of newer applications. It required retooling of network protocols within WSNs [32].

New IPv6 based standards such as 6LoWPAN and ZigBee IP were proposed for WSN in order to introduce flexibility and easy interoperability with computer

networks. The problems in integration of IP with WSN may be seen from two different aspects:

1. Differences in the communication pattern and capabilities WSN and IP, and
2. Expected operational capabilities of WSNs after integration.

The problems arising out of the differences in communication pattern are due to practical usage scenarios involving WSN deployments. The flow pattern of data in TCP/IP network is between the two ends of a connection. In contrast, the data flow patterns in WSN are either many to one, or one to many. For example, a query may be broadcast from a sink node to all sensors for reporting their individual measured values. Similarly, all sensors having values above the threshold placed by a query should report their respective measurements to the sink node.

The problem arising out of the difference in communication capabilities are due to the variations in standards defining the two protocol stacks: WSN and TCP/IP. Firstly, any non trivial deployment of WSN would consist of hundreds and thousands of nodes. So building and maintaining a global addressing scheme for even a mid sized WSN deployment is not quite easy. It requires prohibitive administrative overheads. Secondly, the ad hoc nature of deploying sensor requires WSNs to be self-configurable. Thirdly, in an IP based network bandwidth requirement for data transfer is high (images, video, sound, etc. can be transferred). This implies IPv4 is not feasible for WSNs. IP header itself too large compared to IEEE 802.15.4 PPDU which is the defined standard for physical layer of sensor nodes. Fourthly, a number of existing WSN deployments do not implement IP stack. The only way to communicate with sensor nodes belonging to such a WSN is possible through specialized gateways. Therefore, the integration IP networks with WSNs not implementing either 6LoWPAN or ZigBee IP is a real concern for interoperability.

WSN protocols work at link layer not at IP layer. MAC protocols for WSN are specially designed for periodic listen and sleep schedule. If the schedules of listen and sleep can be properly planned then it helps to reduce energy wastage and enhance lifetime of WSN. The main sources of energy waste can be attributed to following [33]:

1. Collisions: it not only increases latency but retransmissions also contribute substantially in energy drain out.
2. Overhearing: a node unnecessarily picks up a piece of data not destined for itself.
3. Control packets: using large number of control packets is another source of energy waste.
4. Idle listening: the major source of wastage is wait and listen for packets that may not come. In fact, at least half the time is lost in idle listening, and it may consume energy in excess of 50%.

Considering the energy wastage issue, MAC layer scheduling makes sense.

9.7.1 NAT Based IP-WSN Interconnection

Non IP based solutions to integration of IP with WSN are motivated by two favorable aspects of the whole problem. Firstly, in WSN, each node gather data from environment and reports its observation to a single data sink either directly or may be via multiple hops. So, a WSN effectively operates like an inverted directed tree with base station or the data sink as the root. Secondly, the nodes in WSN are accessible only through base station which acts as a gateway or a proxy. So, a WSN appears like a private network with an independent gateway. Therefore, the most common approach, to integration of WSN and IP, is motivated by NAT semantics for accessing the Internet from private networks.

A WSN operates with IP network through a proxy or gateway. The proxy needs to implement both the protocol stacks: IP and WSN. The scenario of integration is illustrated by Fig. 9.10. Proxy becomes a gateway for communicating with sensor node from IP network. Sending data upstream from sensors to a IP node is not a big problem. Placing a serial forwarder between base station and IP node would serve this purpose [34] as illustrated in Fig. 9.11.

The downstream communication is problematic. Frame structure of two networks are different. A server module can be created specially for providing a NAT like semantic for IP to sensor communication. The problem of sensor not having IP address is solved by treating the sensor network as private network and providing NAT services to address a specific sensor node. A one to one mapping sensor `nodeID`s and `portID`s is created. The server assigns a `portID` to each `nodeID`. For downstream routing, reverse mapping is necessary. So, server module stores the reverse mapping of `portID`s to `nodeID`s. One of the problem with this solution is that the mapping is not static due to variation in propagation pattern of radio signals. Therefore, provision should be made to ensure change in mapping specially regarding change of parents in the tree. The sensor nodes send control packets from time to time to make the server aware of modification in tree structure. Accordingly, the server modifies its information and maintains the WSN tree.

Initially, IP nodes desiring to access data service from a specific WSN are required to register with the respective servers. A server offers this registration service on a

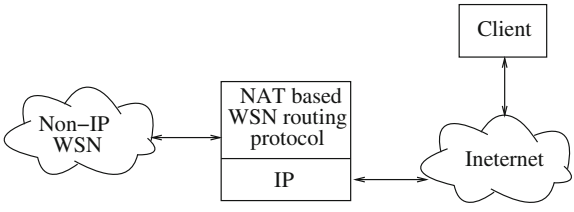

Fig. 9.10 NAT based IP/WSN integration

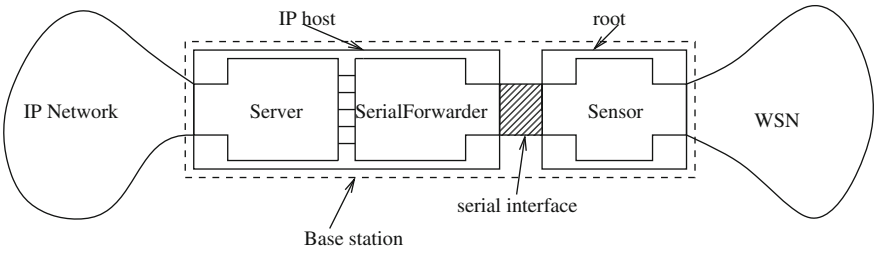

Fig. 9.11 Serial forwarder for WSN to IP communication [34]

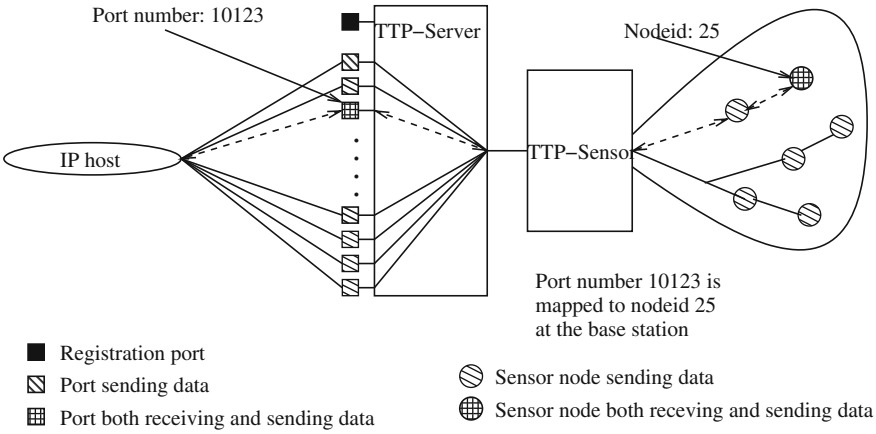

Fig. 9.12 Downstream communication between IP and WSN [34]

known universal port. When data is received from sensor nodes, the same is disseminated to all registered IP hosts as UDP packets. The payload of WSN data packet is sent as payload of UDP packet at a universal port for availing services of downstream communication. The server node stores a mapping of sensor nodes to port numbers. When a registered IP node wants to communicate with a specific sensor node, the node sends the packet addressed to specific port ID. The server module takes care of disseminating information to specific node. Figure 9.12 provides a brief sketch of the process involved.

Source routing is needed for downstream routing. The reason for this is it needs very little support from forwarding node. MAC level ACK is used for retransmission requirements, if any. The server module takes care of packet formation when it receives a downstream packet for a specific sensor node. The server creates the source routing packet with the help of the tree information available to it. The communication is then staged through the serial forwarder. For further details concerning the protocol the reader is referred to [34].

9.8 Data Fusion in WSN

In a data centric network, the communication paradigm is driven by flow of data. Communication is mostly point to multipoint, i.e., either multicast or any cast. The data and its context defines the destination of a data flow. The shift in communication paradigm from address centric to data centric introduces the problem of organizing *data as communication token*. In particular, when network has comparatively large, the number of data generation sources, like in a WSN, the volume of data is unmanageably high. Only meaningful data with assured quality should participate in communication. This implies (i) the accuracy of data, and (ii) the elimination of unwanted data should be included among the important parameters for the definition of the quality of data. Data fusion in WSN primarily achieve the aforesaid objectives. Before going deeper into data fusion techniques, let us examine the question: why data fusion is important in the context of WSNs?

Sensors measure raw data, for example, temperature, pressure, humidity, luminosity, radiation levels, air pollutants, etc. There may be errors in these measurements due to various reasons including the sensors being erroneous. Furthermore, these raw data make little sense unless they are processed and interpreted. So, the key to use of WSN is the interpretation of the sensed raw data. Typically the sensed data received from multiple data sources are combined to improve the quality and the efficacy of interpretation. Apart from eliminating errors, combining data from multiple sources also helps to eliminate sensor failures, temporal and spatial coverage problems. Sensor fusion is particularly important for supporting applications for smart environments. For example, in smart health care system the vital parameters of a critical patient are monitored on continuous basis, and alarms should be raised when these parameters exhibit abnormalities. If the measurements are not accurate, then the patient's life could be endangered. Similarly, if the radiation levels in operations of a nuclear power plant are not correctly monitored it can lead to disastrous consequences. In summary, the sensor equipped monitoring systems lead to the detection of events which sets in motions actuation processes to enhance the ability of a system to respond smartly. Data fusion can be seen as the process of combining data collected from multiple sources to enhance the quality of data in a cheaper way and raises its significance.

Simple aggregation techniques like summing, finding minimum or finding maximum have been used to reduce the volume of data in order to minimize the traffic or data flow. Other sophisticated fusion techniques are used for enhancing quality of data. There is some amount of confusion as well as disagreements in understanding the terminology of data fusion [35]. The problem comes from the fact that the terminology has been developed out of various systems, architectures, applications, methods, theories about data fusion [35]. There is no unified terminology for description of data fusion. Nakamura et al. [35] mention many different ways to describe general term of data and information fusion.

9.8.1 Definitions

Let us discuss a few of these definitions just to understand why lot of confusions surround the concept of data diffusion. The first definition is from the lexicon of US Department of Defense [36]. It says:

Definition 9.1 Data fusion is a multi-level and multifaceted process dealing with automated detection, association, correlation, estimation and combination of data and information from multiple sources.

The above definition does not restrict itself to sensory data alone. Data can be sourced from a variety of other sources such as satellites, remote sensing equipments, robotics, etc. The objective of the second definition proposed in [37] is to capture the improved accuracy of data. According to this definition:

Definition 9.2 Data fusion is combination of data from multiple sensors, and related information provided by associated databases, to achieve improved accuracy and more specific inferences than could be achieved by the use of a single sensor alone.

The third definition is developed with an objective of describing enhanced quality of data [38] in the context of application or applications it is intended to serve. It defines:

Definition 9.3 Data fusion is a formal framework in which are expressed means and tools for the alliance of data originating from different sources. It aims at obtaining information of greater quality; the exact definition of *greater quality* will depend upon the application.

The fourth one is a verbose definition used in the context of data fusion in robotics and vision. It tries to capture both the use of fusion data, and its interaction with the environment. It prefers the term multisensor integration over data fusion. According to this definition multisensor integration [39] is:

Definition 9.4 The synergistic use of information provided by multiple sensory devices to assist in the accomplishment of a task by a system; and multisensor fusion deals with the combination of different sources of sensory information into one representational format during any stage in the integration process.

However, among the community of WSN researchers, fusion is typically seen as an aggregation of data [40–42]. In some sense, the definition of data aggregation appears to refer to composition of raw data into piece of information that can be used for filtering unwanted information and also to summarize the contents. Accordingly, data aggregation [40] is defined as follows.

Definition 9.5 Data aggregation comprises the collection of raw data from pervasive data sources, the flexible, programmable composition of the raw data into less voluminous refined data, and the timely delivery of the refined data to data consumers.

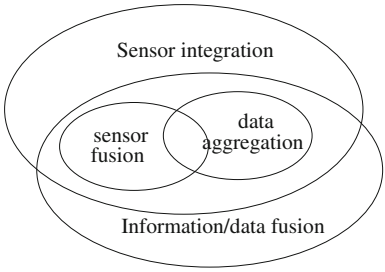

Fig. 9.13 Data/information fusion [35]

Figure 9.13 [35] illustrates the subtle differences that exist among multisensor integration, data aggregation, sensor fusion and information/data fusion. Information essentially refers to processed or organized data. However, certain data may already be organized in some way before it gets accessed by destinations. So, two terms at times refer to the same concept and can be used interchangeably. In a coarse granularity, sensor fusion and data aggregation encompassed within the both sensor integration and data/information fusion. Multisensor integration relates to fusion of data only from sensors. So, it is restrictive in general context of data fusion. Though sensor fusion can be just aggregation of data from sensors, it can include more elaborate form of data processing. Similarly, the generic meaning of data aggregation does not just relate to aggregation of data collected from sensors alone.

9.8.2 Data Collection Model

The key design issues in sensor data diffusion are:

1. How do the interests to acquire sensor data get disseminated?
2. How do the sensor nodes keep track of different interests?
3. How do the nodes propagate data when subscribed events occur?
4. How do the nodes apply data processing (aggregation/fusion) as the data get propagated towards the sink?

The interests are expressed on the basis of the attributes. For example, if the sensors are deployed to monitor forest fire, then an interest can be expressed as follows:

$$\{Type = Fire, Subtype = Amber, Interval = 2\,\text{m},$$
$$Time = 1\,\text{h}, Region = [-20, 20, 300, 200]\}$$

The interest gets disseminated to all sensors belonging to region specified by the bounding box $[-20, 20, 300, 200]$. The sensors are supposed to monitor the fire. The fire can be of different Subtypes, e.g., subtype *Amber* could mean that a forest fire breakout is predicted. Other subtypes could be *Red*, or *Black* meaning that fire is now on, and may have disastrous consequences. The time parameters instructs the sensor

to send observations for the next 1 h in intervals of 2 m. The interests are flooded into the network. All the sensor nodes with matching data commence reporting to data sink from which the interest originated.

Now, let us examine how a sensor node reports when an event occurs? For example, suppose a sensor node sensed a high temperature in its vicinity, it could create a tuple of the form:

$$\{Type = Heat, Subtype = VeryHot, Temperature = 200,$$
$$Location = [101, 205], Timestamp = 14 : 50 : 05, Confidence = 0.7\}$$

It includes three key items of information, namely, location of the source, timestamp and confidence value apart from the attribute values. So, the data sink can analyze the occurrences of the events.

The diffusion starts with initial expression of interests for data by the sink. The interest may be viewed as a subscription for the events in a broader framework of a publish subscribe model. The interest and the arrival of data are combined cleverly to set up a gradient for the data to flow. The process is described as follows:

- The initial interest is essentially an exploration phase. During this phase, the data sink sets a larger interval for the data sampling by the sensor nodes.
- The second phase is concerned with the creation of gradient and its reinforcement. After receiving the first few data items related to interest, the sink refreshes the interest by using the reverse path. In the refreshed interests, the sink lowers the sampling interval for the sensor nodes.

In exploration phase, the interests are sent with larger interval primarily to reduce flooding. The interest get disseminated by flooding.

The processing of interest dissemination works as follows

1. Each node creates an interest cache.
2. When a node receives an interest, it checks if the interest has an entry in the local interest cache.
3. If no entry exists, then the node caches the interest and also creates a reverse gradient by listing the neighbor who sent the interest.
4. It then forwards the interest to its other neighbors excluding the neighbor from which it received the interest.

Many routes may be created in the reverse direction (from sources to sink) for the flow of data. If the flow of data is allowed with high frequency from the sources, it could lead to huge network traffic even before the best path can be determined. Unnecessary network traffic should be minimized.

The flow of data requires time to settle for the best route as it starts flowing. For settling on the best route, there is no need to use any extra control packets. By making use of flow of data and refreshing interest, the best route can be settled. A gradient is formed when data starts arriving at the sink from the targeted sensors. The data sink tries to reinforce the reverse path through which first few items of data arrived. So,

it sends refreshed interests by lowering the monitoring interval. This way the route settles down with an appropriate gradient and flooding can be restricted.

As far as reporting is concerned, the interval of reporting is the key consideration. There are three possibilities here. The system may depend on a periodic reporting. For example, in the application that requires monitoring of some kind, the sensed data is sent periodically for analysis. In other applications, reporting may be based on the queries. In yet another set of applications event-triggered reporting could be important.

9.8.3 Challenges in Data Fusion

Data fusion addresses four qualitative aspects of the data [43], namely

1. Imperfection,
2. Correlation,
3. Inconsistency, and
4. Disparateness.

Imperfection covers uncertainty, imprecision and granularity. Uncertainty raises doubts about the value of data, while imprecision originates from the measurement related errors. It includes vagueness, ambiguity and incompleteness in the measurement. The degree of data granularity represents its ability to expose level of details.

Correlation exhibit dependencies in data. Fusion of correlated data may create problems unless dependencies are handled carefully. Inconsistencies mean that the same data from different sources do not match. Integrity of data becomes an issue if data is not consistent. Finally, disparateness means data presented from different sources in different format. Some data may be raw, some may be semi-processed and others may be processed. Fusion of such data could be challenging.

Imperfection in data mostly handled by methods having origins in statistical methods. Some of this methods are based on probability theory [44], rough set theory [45], fuzzy set [46], Dempster-Shafer theory [47]. Probability theory is used to handle uncertainty. Fuzzy set used to remove vagueness. Evidential and belief theory (Dempster-Shafer) is used to eliminate both uncertainty and ambiguity in data. Rough sets are used for granularity problem.

9.8.4 Data Fusion Algorithms

There are many ways data fusion can be performed. Classification of fusion techniques can be made on the basis of various criteria such as (i) data abstraction level (ii) purpose, (iii) parameters (iii) types of data, and (iv) mathematical foundations. Nakamura et al. [35] have presented an excellent comprehensive review on the topic. The contents, presented here, have been drawn largely from the above paper. But

our focus is limited to only a few algorithms designed on the basis of interesting mathematical framework.

According to [35], the purpose of applying a fusion algorithm can be the following:

1. To derive an inference that can assist in making a decision.
2. To estimate certain value or a vector of values from the sensor observations.
3. To determine the features representing aspects of an environment from the raw sensory data.
4. To compute reliable abstract sensors which can be used for time synchronization so that the sensors can maintain upper and lower bounds on the current time.
5. To aggregate sensor data for the purpose of overcoming the problem of implosion and overlap of data (explained earlier in 9.4.1).
6. To compress similar data by exploiting spatial correlation among sensor data.

Apart from the classes of algorithms mentioned above, there has been some attempts to tackle data fusion problem based on information theory [48]. The purpose is to increase the reliability by fusion of data from multiple sensors. For example, it may be possible to correlate observations from multiple sensing capabilities such as acoustic, magnetic, luminosity, etc. It increases reliability as well confidence in sensor observations. This class of algorithms are still in infancy. They need to overcome the difficulties such as uncertainties in measurements of entropies of source nodes, and efficient conversion of observations to entropies.

9.8.4.1 Bayesian Inference and Dempster-Shafer Belief Model

Let us examine two inference models which have been used extensively indata fusion area [35]. The first one is the Bayesian inferencing model, where the uncertainty is represented by conditional probabilities describing the belief. It is based on the well known and classical Bayes' rule [49]:

$$Pr(Y|X) = \frac{Pr(X|Y)Pr(Y)}{Pr(X)},$$

where the probability $Pr(Y|X)$ represents the belief of hypothesis Y with known information X. It essentially states that the probability of Y given X is obtained by multiplying $Pr(Y)$ with the probability $Pr(X|Y)$ of X given Y is true, with $Pr(X)$ being a normalizing constant. The problem with Bayesian rule is it requires either the exact or near approximate guess of probabilities $Pr(X)$ and $Pr(X|Y)$. In WSN, Bayesian inference rules have been used mostly for localization problem based on fusing information from the mobile beacons [50].

Dempster-Shafer theory [47] is a mathematical tool for combining independent evidence components in order to arrive at a decision. It determines the degrees of belief for one question from the subjective probabilities of a related question. This theory is extensively applied in the expert system [51]. The main features of this theory are:

- Each fact has a degree of support belonging to interval [0, 1], where
  - 0 represents no support,
  - 1 represent full support.
- A set of possible conclusions is represented by

$$\Theta = \{\theta_1, \theta_2, \ldots, \theta_n\}$$

  where:

  - Each θ_i is mutually exclusive, i.e., only one can hold.
  - Θ is exhaustive, i.e., at least one θ_i must hold.

Dempster-Shafer theory is concerned with the evidences that support subsets of outcomes in Θ. The power set of Θ is defined as the *frame discernment* which represents all possible outcomes. For example, if $\Theta = \{a, b, c\}$ then the frame of discernment is given by:

$$\phi, \{a\}, \{b\}, \{c\}, \{a, b\}, \{b, c\}, \{c, a\}, \{a, b, c\}$$

The empty set ϕ has probability 0. Also $\{a, b, c\}$ has probability 0, because by assumption only one of the outcomes would hold, not all.

Furthermore, it defines four different measurable functions:

- Mass function: For a $A \subseteq P(\Theta)$ mass function $m(A)$ is simply the proportion of all the evidences that support A. The mass expresses all the relevant evidences that supports the claim that actual state belongs to A and not to any particular subset of A.
- Belief function: The belief in $A \subseteq P(\Theta)$ is defined as the sum of masses of an element which are subsets of A including itself. For example, if $A = \{a, b, c\}$, then

$$bel(A) = m(a) + m(b) + m(c) + m(a, b) + m(b, c) + m(c, a) + m(a, b, c)$$

- Plausibility function: Plausibility of A, $pl(A)$ is the sum of all the mass of the sets that intersect with A.
- Disbelief function. Disbelief or doubt in A is simple $bel(\neg A)$. It is calculated by summing all mass values of elements that do not intersect A.

Dempster-Shafer theory associates a certainty with a given subset A by defining a belief interval which is $[bel(A), pl(A)]$. The belief interval of a set A defines the bounds within which probability of outcome A lies. Dempster-Shafer theory also defines a composition rule for combining effects of two basic probability masses:

$$m_1 \oplus m_2(\phi) = 0$$
$$m_1 \oplus m_2(A) = \frac{\sum_{x \cap Y = A} m_1(X) m_2(Y)}{1 - \sum_{X \cap Y} m_1(X) m_2(Y)}$$

Dempster-Shafer theory has been used in sensor data fusion because it is considered to be more flexible than Bayesian inference rule. It allows each source to contribute different levels of details to information gathering [35]. For example, to quote an example from [35], suppose there are two different sensors S_1 and S_2. Suppose S_1 can recognize the roar of a male feline, while S_2 can recognize the roar of a female feline. Now a third sensor, S_3 is also there that can distinguish between the roars of a Cheetah and a Lion. Dempster-Shafer theory allows us to combine the data from three sensors to conclude whether the recorded roar is that of a male/female Cheetah. Dempster-Shafer theory does not assign a priori probabilities to unknown events unlike Bayesian model. The probabilities are assigned only when supporting facts are available.

9.8.4.2 Reliable Abstract Sensors

Marzullo [52] proposed the idea of a reliable abstract sensor for arriving at an interval that will always contain the actual value of a desired physical state variable. It has been used in the context of time synchronization, wherein the sensors can perform external synchronization by maintaining the upper and lower bound on the current time.

The term reliable abstract sensor has been used to define one of the three sensors: concrete sensor, abstract sensor and reliable abstract sensor. A concrete sensor is a physical sensor, an abstract sensor is an interval of value representing the measurement of a state variable provided by a concrete sensor. A reliable abstract sensor is the interval that assuredly always contain the real value of the physical variable. So, abstract sensors are mathematical entities representing the concrete sensors.

Marzullo [53] proposes a fault tolerant averaging algorithm for agreement in distributed system. This algorithm is the basis for NTP [54]. Subsequently Marzullo [52] adopted the same algorithm for reliable abstract sensors. It assumes that at most f out of $n \geq 2f + 1$ sensors can be faulty. Let $\mathscr{I} = \{I_1, I_2, \ldots, I_n\}$ be the set of intervals produced by abstract sensors. The idea is to compute $M_n^f = [low, high]$ by fault tolerant averaging, i.e.,

1. low is the smallest value belonging to at least $n - f$ intervals,
2. $high$ is the largest value belonging to at least $n - f$ intervals.

The algorithm computes the fault tolerant averaging in $O(n \log n)$. Clearly, M_I^f cannot give more accurate value than the most accurate sensor when $n = 2f + 1$. Also any minor changes in input could produce a very different output. In other words, the result is unstable. To appreciate the implication of fault tolerant averaging consider an example illustrated by Fig. 9.14. It represents intervals I_1, I_2, I_3 and I_4 reported by four sensors S_1, S_2, S_3 and S_4, respectively. Let one of these four sensors be faulty. Since, I_2 and I_3 do not intersect, one of S_2 and S_3 must be faulty. Marzullo's interval overlapping method produces the intervals M_I^f as shown in Fig. 9.14. It produces unstable output, as can be checked by shifting the interval I_3 to right.

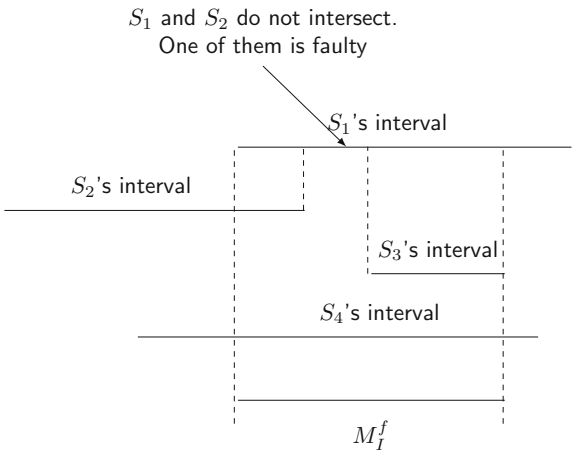

Fig. 9.14 Marzullo's fusion technique with reliable sensor [52]

9.8.4.3 Compression

Distributed Source Coding (DSC) is a compression technique for correlated data from multiple sources [55]. The sources are distributed and do not communicate for coding. The compressed data is sent to a central sink node for decoding. Kusuma [56] and Pradhan et al. [57] proposed Distributed Source Coding Using Syndromes (DISCUS) framework for data compression.

To understand the coding, let us consider an example. Suppose the observations from sensors are coded in a 3-bit word. So, the possible observations are {000, 001, 010, 011, 100, 101, 110, 111}. These observations are grouped into 4 cosets whose elements have hamming distance of three, i.e.,

1. Coset 1: {000, 111} with index 00
2. Coset 2: {001, 110} with index 01
3. Coset 3: {010, 101} with index 10
4. Coset 4: {011, 100} with index 11.

A node sends its observation by sending only the corresponding index. A node S_2 can decode S_1's observation by the fact that hamming distance between its own observation and the observation of S_1 is 1. For example, if S_1's observation is 101, it sends 10 to S_2. Let S_2's observation be 101, then S_2 and can decode S_1's observation from the index 10 and the fact that hamming distance between two observations is 1. The overall framework of data compression by DISCUS framework is illustrated by Fig. 9.15.

The data fusion is not just limited to sensor fusion or multisensor integration. It refers to the larger context of fusion of data from various source as in robotics, remote sensing, etc. So, the data fusion techniques have their origin even before wireless sensor were deployed. But interestingly, it was possible to adopt many of the techniques of fusion to multisensor integration. In this chapter, the scope of

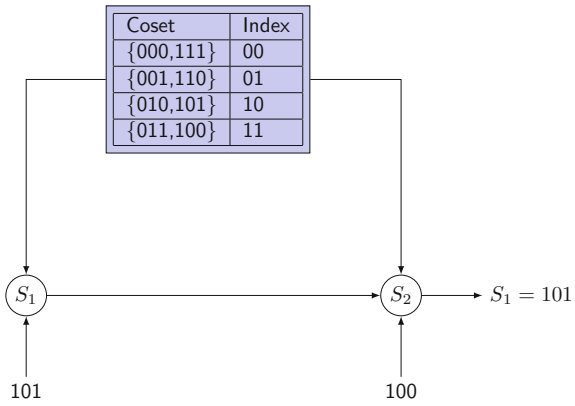

Fig. 9.15 Data compression in DISCUS [35]

discussion is limited to the extent of exposing mathematical richness of a few of these techniques to the reader. However, sufficient pointers to extensive literature have been provided for an interested reader to follow up further readings.

References

1. S. Distefano, G. Merlion, A. Puliafito. Sensing and actuation as a service: a new development for clouds, in *The 11th IEEE Internation Symposium on Network Computing and Applications* (2012), pp. 272–275
2. G. Anastasi, M. Conti, M. Di Francesco, A. Passarella, Energy conservation in wireless sensor networks: a survey. Ad hoc Netw. **7**(3), 537–568 (2009)
3. K.C. Barr, K. Asanović, Energy-aware lossless data compression. ACM Trans. Comput. Syst. (TOCS) **24**(3), 250–291 (2006)
4. G.J. Pottie, W.J. Kaiser, Wireless integrated network sensors. Commun. ACM **43**(5), 51–58 (2000)
5. S. Choi, N. Kim, H. Cha, Automated sensor-specific power management for wireless sensor networks, in *5th IEEE International Conference on Mobile Ad Hoc and Sensor Systems* (IEEE Computer Society, Atlanta, GA, USA. Atlanta, GA, USA, 2008), pp. 305–314
6. H. Kim, H. Cha, Towards a resilient operating system for wireless sensor networks, in *USENIX Annual Technical Conference, General Track* (2006), pp. 103–108
7. X. Fang, S. Misra, G. Xue, D. Yang, Smart gridthe new and improved power grid: a survey. IEEE Commun. Surv. Tutor. **14**(4), 944–980 (2012)
8. T. Hikihara, Power router and packetization project for home electric energy management, in *Santa Barbara Summit on Energy Efficiency* (2010), pp. 12–13
9. J.M. Kahn, R.H. Katz, K.S.J. Pister. Next century challenges: mobile networking for "smart dust", in *Mobicom'99* (Seattle, Washington, USA, 1999), pp. 271–278
10. M.O. Farooq, T. Kunz, Operating systems for wireless sensor networks: a survey. *Sensors***11**(6), 59005930 (2011) (Basel, Switzerland)
11. P. Levis, S. Madden, J. Polastre, R. Szewczyk, K. Whitehouse, A. Woo, D. Gay, J. Hill, M. Welsh, E. Brewer, D. Culler, TinyOS: an operating system for sensor network, in *Ambient intelligence* (Springer, Berlin Heidelberg, 2005), pp. 115–148
12. A. Dunkels, B. Gronvall, T. Voigt, Contiki-a lightweight and flexible operating system for tiny networked sensors, in *29th Annual IEEE International Conference on Local Computer Networks* (IEEE, 2004), pp. 455–462

13. J. Wang, R.K. Ghosh, S.K. Das, A survey on sensor localization. J. Control Theory Appl. **8**(1), 2–11 (2010)
14. J.N. Al-Karaki, A.E. Kamal, Routing techniques in wireless sensor networks: a survey. IEEE Wirel. Commun. **11**(6), 6–28 (2004)
15. W. Heinzelman, J. Kulik, H. Balakrishnan, Adaptive protocols for information dissemination in wireless sensor networks, in *The 5th ACM/IEEE Mobicom Conference (MobiCom 99)* (Seattle, WA, 1999), pp. 174–85
16. W. Heinzelman, A. Chandrakasan, H. Balakrishnan, Energy-efficient communication protocol for wireless microsensor networks, in *The 33rd Hawaii International Conference on System Sciences (HICSS 00)*, January 2000
17. N. Bulusu, J. Heidemann, D. Estrin, GPS-less low cost outdoor localization for very small devices. Technical report, University of Southern California, April 2000. Technical report 00-729
18. S. Capkun, M. Hamdi, J. Hubaux, GPS-free positioning in mobile ad-hoc networks, in *The 34th Annual Hawaii International Conference on System Sciences (HICSS'01)* (2001), pp. 3481–3490
19. A. Savvides, C.C. Han, M. Srivastava, Dynamic fine-grained localization in ad-hoc networks of sensors, in *The Seventh ACM Annual International Conference on Mobile Computing and Networking (MobiCom)*, pp. 166–179, July 2001
20. B. Chen, K. Jamieson, H. Balakrishnan, R. Morris, SPAN: an energy-efficient coordination algorithm for topology maintenance in ad hoc wireless networks. Wirel. Netw. **8**(5), 481–494 (2002)
21. Y. Xu, J. Heidemann, D. Estrin, Geography-informed energy conservation for ad-hoc routing, in *The Seventh Annual ACM/IEEE International Conference on Mobile Computing and Networking* (2001), pp. 70–84
22. Y. Yu, D. Estrin, R. Govindan, Geographical and energy-aware routing: a recursive data dissemination protocol for wireless sensor networks. Technical report, University of California at Los Angeles, May 2001
23. J.H. Chang, L. Tassiulas, Maximum lifetime routing in wireless sensor networks, in *Advanced Telecommunications and Information Distribution Research Program (ATIRP), College Park* (MD, USA, March, 2000), p. 2000
24. S. Dulman, T. Nieberg, J. Wu, P. Havinga, *Trade-off between Traffic Overhead and Reliability in Multipath Routing for Wireless Sensor Networks* (In WCNC Workshop, New Orleans, Louisiana, USA, 2003)
25. D. Ganesan, R. Govindan, S. Shenker, D. Estrin, Highly-resilient, energy-efficient multipath routing in wireless sensor networks. ACM SIGMOBILE Mob. Comput. Commun. Rev. **5**(4), 1125 (2001)
26. C. Intanagonwiwat, R. Govindan, D. Estrin, Directed diffusion: a scalable and robust communication paradigm for sensor networks, in *ACM MobiCom 00* (Boston, MA, 2000), pp. 56–67
27. Q. Li, J. Aslam, D. Rus. Hierarchical power-aware routing in sensor networks, in *The DIMACS Workshop on Pervasive Networking*, May 2001
28. C. Rahul, J. Rabaey, Energy aware routing for low energy ad hoc sensor networks, in *IEEE Wireless Communications and Networking Conference (WCNC)*, vol 1 (Orlando, FL, USA), pp. 350–355. 17–21 March 2002
29. D. Braginsky, D. Estrin, Rumor routing algorithm for sensor networks. In *International Conference on Distributed Computing Systems (ICDCS'01)*, November 2001
30. J. Kulik, W.R. Heinzelman, H. Balakrishnan, Negotiation-based protocols for disseminating information in wireless sensor networks. Wirel. Netw. **8**, 169–185 (2002)
31. K. Sohrabi, J. Pottie, Protocols for self-organization of a wireless sensor network. IEEE Person. Commun. **7**(5), 16–27 (2000)
32. A.P. Castellani, N. Bui, P. Casari, M. Rossi, Z. Shelby, M. Zorzi, Architecture and protocols for the internet of things: a case study, in *8th IEEE International Conference on Pervasive Computing and Communications (PERCOM Workshops)* (IEEE, 2010), pp. 678–683

33. W. Ye, J. Heidemann, D. Estrin, An energy-efficient MAC protocol for wireless sensor networks, in *The 21st International Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM), New York* (USA, June, NY, 2002), p. 2002
34. S. Shekhar, R. Mishra, R.K. Ghosh, R.K. Shyamasundar, Post-order based routing and transport protocol for wireless sensor networks. Pervasive Mob. Comput. **11**, 229–243 (2014)
35. E.F. Nakamura, A.A. Loureiro, A.C. Frery, Information fusion for wireless sensor networks: methods, models, and classifications. ACM Comput. Surv. **39**(3) (2007)
36. F.E. White, Data fusion lexicon. Technical report, U.S. Department of Defense, *Code 4202* (NOSC, San Diego, CA, 1991)
37. D.L. Hall, J. Llinas, An introduction to multi-sensor data fusion. Proceedings of IEEE **85**(1), 6–23 (1997)
38. L. Wald, Some terms of reference in data fusion. IEEE Trans. Geosci. Remote Sens. **13**(3), 1190–1193 (1999)
39. R.C. Luo, M.G. Kay (eds.), *Multisensor Integration and Fusion for Intelligent Machines and Systems* (Ablex Publishing, New Jersey, USA, 1995)
40. N.H. Cohen, A. Purakayastha, J. Turek, L. Wong, D. Yeh. Challenges in flexible aggregation of pervasive data. Technical report, IBM Research Division, Yorktown Heights, NY, USA, January 2001. IBM Research Report RC 21942 (98646)
41. K. Kalpakis, K. Dasgupta, P. Namjoshi, Efficient algorithms for maximum lifetime data gathering and aggregation in wireless sensor networks. Comput. Netw. **42**(6), 697–716 (2003)
42. R. Van Renesse, The importance of aggregation, in *Future Directions in Distributed Computing: Research and Position Papers*, ed. by A. Schiper, A.A. Shvartsman, H. Weatherspoon, B.Y. Zhao, vol NCS 2584 (Springer, Bologna, Italy, 2003), pp. 87–92
43. B. Khaleghi, A. Khamis, O. Karray, Multisensor data fusion: a review of the state-of-the-art. em. Inf. Fus. **14**(1), 28–44 (2013)
44. H.F. Durrant-Whyte, T.C. Henderson, Multisensor data fusion, in *Handbook of Robotics*, ed. by B. Siciliano, O. Khatib (Springer, 2008), pp. 585–610
45. Z. Pawlak, *Rough Sets: Theoretical Aspects of Reasoning about Data* (Kluwer Academic Publishers, Norwell, MA, USA, 1992)
46. A. Zadeh, Fuzzy sets. Inf. Control **8**(3), 338–353 (1965)
47. G. Shafer, *A Mathematical Theory of Evidence*. Princeton University Press, 1976
48. P.K. Varshney, *Distributed Detection and Data Fusion* (Springer, New York, USA, 1967)
49. T.R. Bayes, An essay towards solving a problem in the doctrine of chances. Philosop. Trans. R. Soc. **53**, 370–418 (1763)
50. M.L. Sichitiu, V. Ramadurai, Localization of wireless sensor networks with a mobile beacon, in *The 1st IEEE International Conference on Mobile Ad Hoc and Sensor Systems (MASS 2004)* (IEEE, Fort Lauderdale, FL, USA, 2004), pp. 174–183
51. P.P. Shenoy, Using dempster-shafer's belief-function theory in expert systems, in *Advances in the Dempster-Shafer Theory of Evidence*, ed. by R.R. Yager, J. Kacprzyk, M. Fedrizzi (John Wiley & Sons, Inc., New York, NY, USA, 1994), pp. 395–414
52. K. Marzullo, Tolerating failures of continuous-valued sensors. ACM Trans. Comput. Syst. (TOCS) **8**(4), 284–304 (1990)
53. K. Marzullo, Maintaining the time in a distributed system: an example of a loosely-coupled distributed service. PhD thesis, Stanford University, Department of Electrical Engineering, Stanford, CA, 1984
54. D.L. Mills, *Computer Network Time Synchronization: The Network Time Protocol* (Taylor & Francis, 2011)
55. Z. Xiong, A.D. Liveris, S. Cheng, Distributed source coding for sensor networks. IEEE Signal Process. Mag. **21**(5), 80–94 (2004)
56. J. Kusuma, L. Doherty, K. Ramchandran, Distributed compression for sensor networks, in *The 2001 International Conference on Image Processing (ICIP-01)*, vol 1 (IEEE, Thessaloniki, Greece, 2001), pp. 82–85
57. S.S. Pradhan, K. Ramchandran, Distributed source coding using syndromes (DISCUS): design and construction. IEEE Trans Inf Theory **49**(3), 626–643 (2003)