ELECTIVE: IV

BEIT804T4 **WIRELESS SENSOR NETWORKS**

(Theory Credit: 05)

Teaching Scheme:
Lecture: 4 Hours/week
Tutorial: 1 Hour/week

Examination Scheme:
Theory: T (U): 80 Marks T (I): 20 Marks
Duration of University Exam. : 03 Hours

==========

UNIT I:
Introduction to wireless Sensor Network:
Network Characteristics, Network application, Network design challenges, Sensor network architectural elements, WSN standards, IEEE 802.15.4, Zig-bee.

UNIT II:
Basic Wireless Sensor Technology:
Sensor node structures, Sensor network architecture, Classification of WSN, Protocol Stack for WSN.

UNIT III:
Medium Access Control:
Fundamental MAC Protocol, MAC design for WSN, S-MAC, DS-MAC, MS-MAC, Traffic adaptive medium access, Self organizing MAC.

UNIT IV:
Routing in WSN:
Data dissemination and gathering, Routing challenges and design issues in WSN, Routing strategies, Flooding and it's variants, Low energy adaptive clustering, Geographical routing.

UNIT V:
Transport Protocol:
Traditional transport protocol, Transport protocol design, Authenticity: Message authentication code, Signature, Authenticating public key, Broadcast and Multicast authentication.

UNIT VI:
Network Management and Operating System for WSN:
Traditional network management models, network management design issues, Example of management architecture: MANNA, Operating system design issues, Operating System: Tiny OS, Mate OS, Magnet OS.

WSN

Unit 1

1 Network Characteristics of a wireless Sensor Network

Answer:- A WSN typically consists of a large number of low - cost, low - power, and multifunctional sensor nodes that are deployed in a region of interest.

• ***Dense Node Deployment.*** Sensor nodes are usually densely deployed in a fi eld of interest. The number of sensor nodes in a sensor network can be several orders of magnitude higher than that in a MANET.

• ***Battery*** - *Powered Sensor Nodes.* Sensor nodes are usually powered by battery. In most situations, they are deployed in a harsh or hostile environment, where it is very diffi cult or even impossible to change or recharge the batteries.

• ***Severe Energy, Computation, and Storage Constraints.*** Sensor nodes are highly limited in energy, computation, and storage capacities.

• ***Self*** - *Confi gurable.* Sensor nodes are usually randomly deployed without careful planning and engineering. Once deployed, sensor nodes have to autonomously confi gure themselves into a communication network.

• ***Application Specifi c.*** Sensor networks are application specifi c. A network is usually designed and deployed for a specifi c application. The design requirements of a network change with its application.

• ***Unreliable Sensor Nodes.*** Sensor nodes are usually deployed in harsh or hostile environments and operate without attendance. They are prone to physical damages or failures.

• ***Frequent Topology Change.*** Network topology changes frequently due to node failure, damage, addition, energy depletion, or channel fading.

• ***No Global Identifi cation.*** Due to the large number of sensor nodes, it is usually not possible to build a global addressing scheme for a sensor network because it would introduce a high overhead for the identifi cation maintenance.

Source Node → It is a node/sensor where data is collected communication to be place between source & sink

Sink → Sink is nothing but destination or base station used to collect & process data in centralized.

• ***Many - to - One Traffi c Pattern.*** In most sensor network applications, the data sensed by sensor nodes fl ow from multiple source sensor nodes to a particular sink, exhibiting a many - to - one traffi c pattern.

• ***Data Redundancy***. In most sensor network applications, sensor nodes are densely deployed in a region of interest and collaborate to accomplish a common sensing task. Thus, the data sensed by multiple sensor nodes typically have a certain level of correlation or redundancy.

Data Redundancy → same data is held in two ⊖ seperate places.

2. Network application → WSN Application

Sensors can be used to detect or monitor a variety of physical parameters or conditions for example,

- Light
- Sound
- Humility
- Pressure
- Temperature
- Soil composition
- Air or water quality
- Attributes of an object such as size, weight, position, speed, and direction.

i. ***Environmental Monitoring.***

Environmental monitoring is one of the earliest applications of sensor networks. In environmental monitoring, sensors are used to monitor a variety of environmental parameters or conditions.

Habitat Monitoring. Sensors can be used to monitor the conditions of wild animals or plants in wild habitats, as well as the environmental parameters of the habitats.

Air or Water Quality Monitoring. Sensors can be deployed on the ground or under water to monitor air or water quality. For example, water quality monitoring can be used in the hydrochemistry fi eld. Air quality monitoring can be used for air pollution control.

Hazard Monitoring. Sensors can be used to monitor biological or chemical hazards in locations, for example, a chemical plant or a battlefi eld.

Disaster Monitoring. Sensors can be densely deployed in an intended

region to detect natural or non - natural disasters. For example, sensors can be scattered in forests or revivers to detect forest fires or floods.

ii. ***Military Applications.***

WSNs are becoming an integral part of military command, control, communication, and intelligence (C3I) systems

Battlefi eld Monitoring. Sensors can be deployed in a battlefi eld to monitor the presence of forces and vehicles, and track their movements, enabling close surveillance of opposing forces.

Object Protection.Sensor nodes can be deployed around sensitive objects, for example, atomic plants, strategic bridges, oil and gas pipelines, communication centers, and military headquarters, for protection purpose.

Intelligent Guiding. Sensors can be mounted on unmanned robotic vehicles,tanks, fighter planes, submarines, missiles, or torpedoes to guide them around obstacles to their targets and lead them to coordinate with one another to accomplish more effective attacks or defences.

Remote Sensing.Sensors can be deployed for remote sensing of nuclear, biological, and chemical weapons, detection of potential terrorist attacks,and reconnaissance .

iii. ***Health Care Applications.***

WSNs can be used to monitor and track elders and patients for health care purposes, which can signifi cantly relieve the severe shortage of health care personnel and reduce the health care expenditures in the current health care systems

Behavior Monitoring. Sensors can be deployed in a patient ' s home to monitor the behaviors of the patient. For example, it can alert doctors when the patient falls and requires immediate medical attention. It can monitor what a patient is doing and provide reminders or instructions over a television or radio.

Medical Monitoring. Wearable sensors can be integrated into a wireless body area network (WBAN) to monitor vital signs, environmental parameters, and geographical locations, and thus allow long - term, noninvasive, and ambulatory monitoring of patients or elderly people with instantaneous alerts to health care personal in case of emergency, immediate reports to users about their current health statuses, and real - time updates
of users ' medical records.

iv. ***Industrial Process Control***

In industry, WSNs can be used to monitor manufacturing processes or the condition of manufacturing equipment. For example, wireless sensors can be instrumented to production and assembly lines to monitor and control production processes. Chemical plants or oil refi ners can use sensors to monitor the condition of their miles of pipelines. Tiny sensors can be embedded into the regions of a machine that are inaccessible by humans to monitor the condition of the machine and alert for any failure.

v. ***Security and Surveillance***

WSNs can be used in many security and surveillance applications. For example, acoustic, video, and other kinds of sensors can be deployed in buildings, airports, subways, and other critical infrastructure.

vi. ***Home Intelligence***

WSNs can be used to provide more convenient and intelligent living environments for human beings.

Smart Home. Wireless sensors can be embedded into a home and connected to form an autonomous home network. For example, a smart refrigerator connected to a smart stove or microwave oven can prepare a menu based on the inventory of the refrigerator and send relevant cooking parameters to the smart stove or microwave oven, which will set the desired temperature and time for cooking. The contents and schedules of TV, VCR, DVD, or CD players can be monitored and controlled remotely to meet the different requirements of family members.

Remote Metering. Wireless sensors can be used to remotely read utility meters in a home, for example, water, gas, or electricity, and then send the readings to a remote center through wireless communication .

3. Network design challenges :→

Limited Energy Capacity. Sensor nodes are battery powered and thus have very limited energy capacity. This constraint presents many new challenges in the development of hardware and software, and the design of network architectures and protocols for sensor networks. To prolong the operational lifetime of a sensor network, energy efficiency should be considered in every aspect of sensor network design, not only hardware and software, but also network architectures and protocols.

Limited Hardware Resources. Sensor nodes have limited processing and storage capacities, and thus can only perform limited computational functionalities.

These hardware constraints present many challenges in software development and network protocol design for sensor networks, which must consider not only the energy constraint in sensor nodes, but also the processing and storage capacities of sensor nodes.

Massive and Random Deployment. Most sensor networks consist of a large number of sensor nodes, from hundreds to thousands or even more. Node deployment is usually application dependent, which can be either manual or random. In most applications, sensor nodes can be scattered randomly in an intended area or dropped massively over an inaccessible or hostile region. The sensor nodes must autonomously organize themselves into a communication network before they start to perform a sensing task.

Dynamic and Unreliable Environment. A sensor network usually operates in a dynamic and unreliable environment. On one hand, the topology of a sensor network may change frequently due to node failures, damages, additions, or energy depletion. On the other hand, sensor nodes are linked by a wireless medium, which is noisy, error prone, and time varying. The connectivity of the network may be frequently disrupted because of channel fading or signal attenuation.

Diverse Applications. Sensor networks have a wide range of diverse applications. The requirements for different applications may vary signify - cantly. No network protocol can meet the requirements of all applications. The design of sensor networks is application specific.

4. Sensor network architectural elements :→

Sensor Types and TechnologyA sensor network is composed of a large number of sensor nodes that are densely deployed [1.38,1.39]. To list just a few venues, sensor nodes may be deployed in an open space; on a battlefield in front of, or beyond, enemy lines; in the interior of industrial machinery; at the bottom of a body of water; in a biologically and/or chemically contaminated field; in a commercial building; in a home; or in or on a human body.

Software (Operating Systems and Middleware) To support the node operation, it is important to have open-source operating systems designed specifically for WSNs. Such operating systems typically utilize a component-based architecture that enables rapid implementation and innovation while minimizing code size asrequired by the memory constraints endemic in sensor networks.

Standards for Transport Protocols The goal of WSN engineers is to develop a cost-effective standards-based wireless networking solution that supports low-tomedium data rates, has low power consumption, and guarantees security and reliability. The position of sensor nodes does not have be predetermined, allowing random deployment in inaccessible terrains or dynamic situations; however, this also means that sensor network protocols and algorithms must possess self-organizing capabilities. For military and/or national security applications, sensor devices must be amenable to rapid deployment, the deployment must be supportable in an ad hoc fashion, and the environment is expected to be
highly dynamic.

Routing and Data Dissemination Routing and data dissemination issues deal with data dissemination mechanisms for large-scale wireless networks, directed diffusion , data-centric routing also known as data aggregation , adaptive routing, and other specialized routing mechanism. Routing protocols for WSNs generally fall into three groups: data-centric, hierarchical, and location-based.

Sensor Network Organization and Tracking Areas of interest involving network organization and tracking include distributed group management (maintaining organization in large-scale sensor networks); self-organization, including authentication, registration, and session establishment; and entity tracking: target detection, classification, and tracking. Dynamic sensor allocation (i.e., how to deal with impaired or unreliable sensors and/or how to ''clean'' and query noisy sensors) is
also of interest.

ComputationComputation deals with data aggregation, data fusion, data analysis, computation hierarchy, grid computing (utility-based decision making in wireless sensor networks), and signal processing. We have already mentioned the desire for data-centric protocols that support in-network processing; however, it must be noted that per-node processing by itself is not sufficient: One needs interpretation of
spatially distributed events and data related to those events. The network may be required to handle in-network processing based on the locality of the data, and queries must be directed automatically to the node or nodes that have the best view of the system (environment) in the context of the data queried.

5. Wireless Sensor Network Standards

To facilitate the worldwide development and application of WSNs, there is a need for building a large low - cost market for sensor products in the field. For

this purpose, it is important to specify relevant standards so that sensor products from different manufacturers may interoperate. A lot of efforts have been made and are under way in many standardization organizations in order to unify the market, leading to low - cost and interoperable devices, and avoiding the proliferation of proprietary incompatible network protocols. To a certain extent, the success of WSNs as a technology will largely rely on the success of these standardization efforts.

6. The IEEE 802.15.4 Standard.

- Data rates of 250 kbps, 40 kbps, and 20 kbps.
- Two addressing modes: 16 - bit short and 64 - bit IEEE addressing.
- Support for critical latency devices, for example, joysticks.
- The CSMA - CA channel access.
- Automatic network establishment by the coordinator.
- Fully handshaking protocol for transfer reliability.
- Power management to ensure low - power consumption.
- Some 16 channels in the 2.4 - GHz ISM band, 10 channels in the 915 – MHz band, and 1 channel in the 868 - MHz band.

The physical layer of the IEEE 802.15.4 standard has been specifi ed to coexist with other IEEE standards for wireless networks, for example, IEEE 802.11 (WLAN) and IEEE 802.15.1 (Bluetooth). It features activation and deactivation of the radio transceiver and transmission of packets on the physical medium. It operates in one of the following three license - free bands:

- 868 – 868.6 MHz (e.g., Europe) with a data rate of 20 kbps.
- 902 – 928 MHz (e.g., North America) with a data rate of 40 kbps.
- 2400 – 2483.5 MHz (worldwide) with a data rate of 250 kbps.

The MAC layer provides data and management services to the upper layers. The data service enables transmission and reception of MAC packets over the physical layer. The management services include synchronization, timeslot management, and association and disassociation of devices to the network. Moreover, the MAC layer implements basic security mechanisms.

7. *The ZigBee Standard*

The IEEE 802.15.4 standard only defi nes the physical and MAC layers without specifying the higher protocol layers, including the network and application layers. The ZigBee standard [33] is developed on top of the IEEE 802.15.4 standard and defines the network and application layers. The network layer provides networking functionalities for different
network topologies, and the application layer provides a framework for distributed application development and communication. The two protocol stacks can be combined together to support short - range low data rate wireless communication with battery - powered wireless devices. The potential

applications of these standards include sensors, interactive toys, smart badges, remote controls, and home automation.

The ZigBee protocol stack was proposed at the end of 2004 by the ZigBee Alliance [34] , an association of companies working together to enable reliable, cost - effective, low - power, wirelessly networked, monitoring, and control products based on an open global standard. The fi rst release of ZigBee was revised at the end of 2006, which introduces extensions on the standardization of application profiles and some minor improvements to the network and application layers.

1. Sensor Node structures in Wireless Sensor Network

A Wireless Sensor Network to Monitor Vibrations with In-Network Analysis

- Structure of sensor node

  The sensor node comprises of sensor, microcontroller, and RF transceiver. It is often driven by a battery or energy harvesting system. The sensor generates analogue signals, and an ADC converts the signals. The microcontroller executes a series of algorithms to process the data. All data will be stored in the microcontroller and transmit through an integrated RF transceiver.

- A group of sensor nodes work collaboratively to perform a common application.

- In many WSN applications, the sensor nodes are battery driven and they are often very difficult to recharge or change the batteries.

- Prolonging network lifetime is a critical issue.

- Sensors often have long period between transmissions (e.g., in seconds).

- Thus, a good WSN design needs to be energy efficient.

A sensor node normally consists of four basic components

- A sensing unit
- A processing unit
- A communication unit
- A power unit

2. Sensor network architecture

According to the way that data are collected, WSNs can be classifi ed into three types: homogenous sensor networks, heterogeneous sensor networks, and hybrid sensor networks

i. ***Homogenous Sensor Networks.***

A homogenous network consists of base stations and sensor nodes equipped with equal capabilities, for example, computational power and storage capacity. Data gathering in this type of networks is based on the structure of data dissemination. Flat and hierarchical topologies are two representative structures being widely studied for data dissemination and gathering in homogenous networks

In a flat network, data aggregation is accomplished by data - centric routing where the base station usually transmits a query message to the sensor nodes via flooding, and the sensor nodes that have data matching the query will send response messages back to the base station. The sensor nodes communicate with the base station via multihop routes by using peer nodes as relays. The choice of a particular communication protocol depends on the specific application. Figure 6.1a illustrates the architecture of a flat network. The sensor nodes are assumed to be stationary once being distributed in the targeted area and the collected sensing information is gathered at the base station.

In a hierarchical network, sensor nodes are organized into clusters where the cluster heads serve as simple relays for transmitting the data. Since the cluster heads have the same transmission capacity as the sensor nodes, the minimum requirement on the number of clusters can be derived from the upper bound of the throughput. Higher throughput can be achieved by using clustering at the cost of having extra nodes functioned as cluster heads. Data aggregation in a hierarchical network involves data fusion at cluster heads, which reduces the number of messages transmitted to the base station, and hence improves the energy effi ciency of the network. A typical structure of a hierarchical network is shown in Fig. 6.1 b.

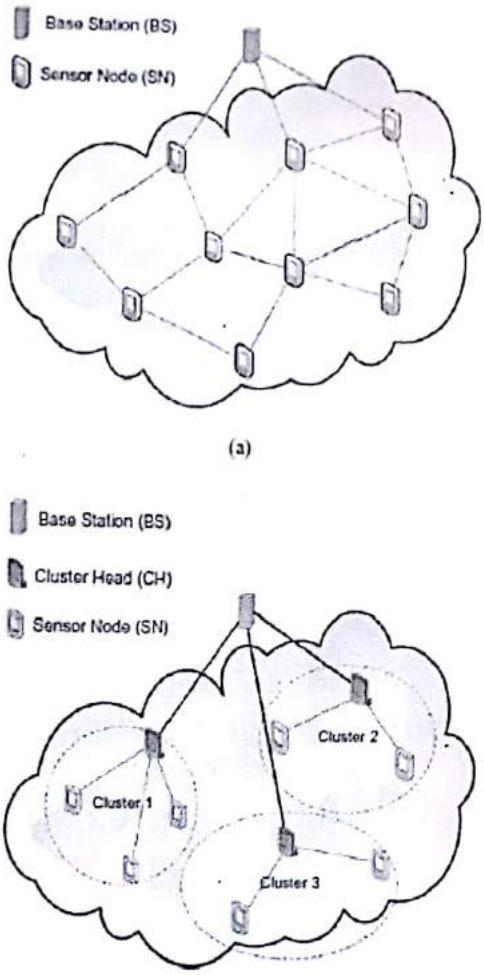

Fig. 6.1 Network topology: (a) flat topology and (b) hierarchical topology.

ii. Heterogeneous Sensor Networks.

A heterogeneous sensor network consists of base stations (fi xed and mobile), sensor nodes, and sophisticated sensor nodes with advanced embedded processing and communicating capabilities as compared to normal sensor nodes. Data gathering can be executed at the mobile base stations. In such networks, mobile base stations move randomly in the area of the deployed network, collecting data directly from normal sensor nodes, or use some surrounding sensor nodes to relay the data (see Fig. 6.2). Sometimes, sensor nodes may be distributed sparsely and the distance between any two sensor nodes can be far apart. The long distance among sensor nodes implies that more energy will be consumed for communication. Meanwhile, sensor nodes need to perform sensing and communication for as long as possible. As shown in many

experimental results, data gathering with mobile sinks is able to prolong the lifetime of the system

iii. Hybrid Sensor Networks.

In a hybrid sensor network, several mobile base stations work cooperatively to provide fast data gathering in a real - time manner. In the scenario shown in Fig. 6.3 , collected data will be relayed by several mobile base stations. The conventional and well - studied routing algorithms for ad hoc networks can be adopted as the routing protocols among these mobile base stations. Mobile ad hoc networks (MANETs) assume that every node is able to move at their own pace. Even though WSNs are more constrained than other wireless networks, for example, MANETs, in terms of energy, processing, transmission range, and bandwidth, routing from a source base station to a destination base station can be accomplished by using MANET protocols in hybrid sensor networks. Hybrid sensor networks can achieve longer lifetime and can also improve the effi ciency of data gathering.

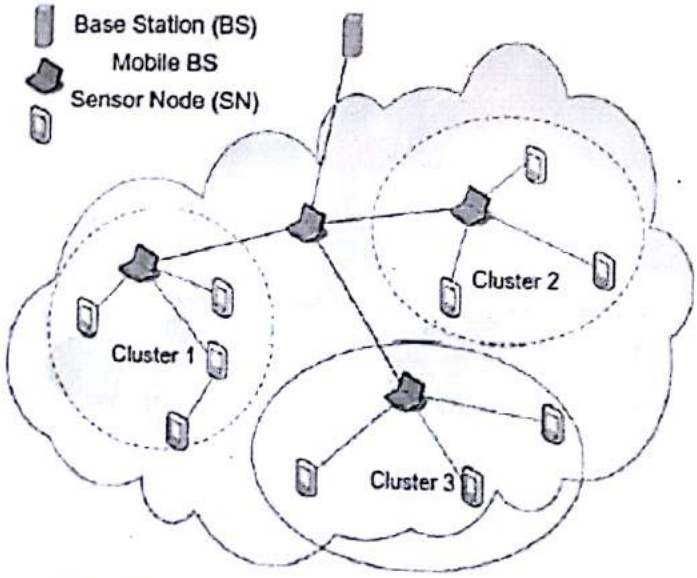

Fig. 6.2 Heterogeneous sensor network topology.

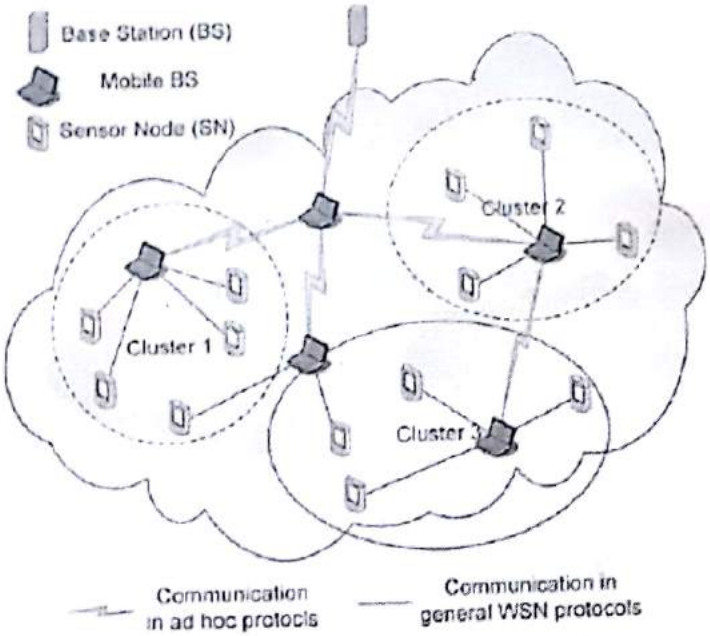

Fig. 6.3 Hybrid sensor network topology.

3. Classification of WSN

A. Group level classification

In this section, we introduce groups' functions in the classification system – collecting, filtering, and aggregating node-level classification results, as well as triangulating the estimated target locations.

Each group has a statically assigned group leader. When events occur, group members periodically report to the group leader. The reports usually consist of node information (e.g., node ID and location), group information (e.g., leader ID and group ID) and even information (e.g., confidence vectors). The group leader aggregates the confidence vectors from group members, computes group-level confidence vectors and reports them to the base-level classification module via multi-hop communication. This scheme greatly reduces the amount of network traffic and, consequently, the energy consumption of the WSN.

Data aggregation contains several tunable parameters that affect different aspects of its performance. One parameter, minimum degree of aggregation (MDOA), defines the minimum number of distinct reports required to form a valid group-level confidence vector. An adequately high MDOA value enhances the credibility of group-level classification results. Hence, it is an important system parameter and has an impact on the performance of the detection and classification.

In the classification system, the groups have a one-to-one mapping to physical events. An implicit assumption is that events always keep a far enough distance among them, so that membership of nodes to the corresponding groups can be determined without ambiguity based on spatial adjacency to one of the events. This results in a limitation on detecting multiple simultaneous targets – for events that become close enough or cross each other, if they share the same sensory signature (e.g., two persons walking together), the current classification system cannot separate them.

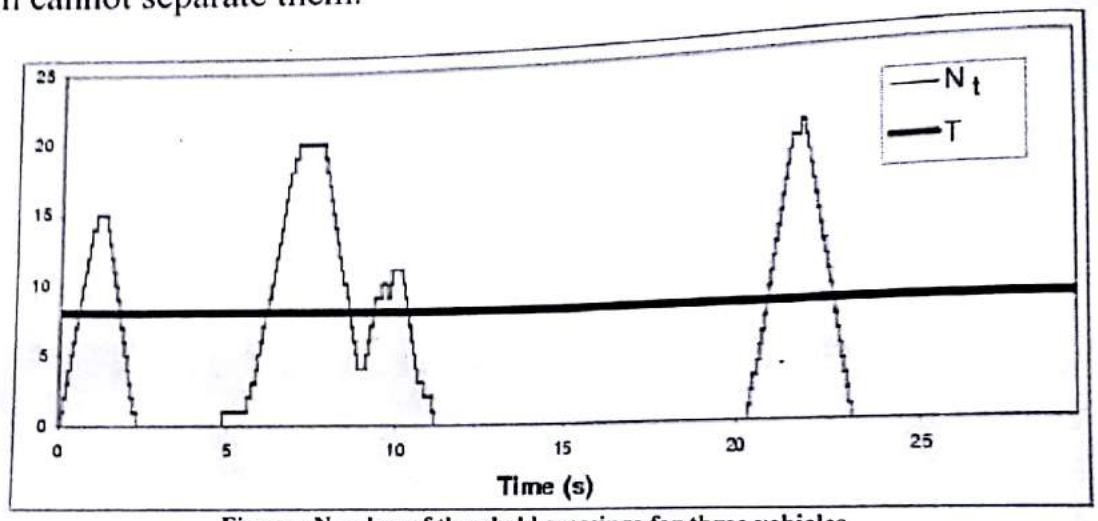

Figure : Number of threshold crossings for three vehicles

B. Base Level Classification

The highest level detection and classification are conducted on the base mote. It takes the group-level classification results as input and computes the final classification results. Since the base mote has a global view of the classification process, it conducts the tasks requiring global knowledge, which is not available to individual nodes or groups. In order to further reduce false positives, spatial and temporal correlations among the tracking reports must be leveraged. Intuitively, the base mote deems that two reports in a certain time frame are from the same target if their locations are close. The base mote keeps a history of recently received reports. With the history of reports, the classification and velocity calculation of each target can be accomplished with high accuracy.

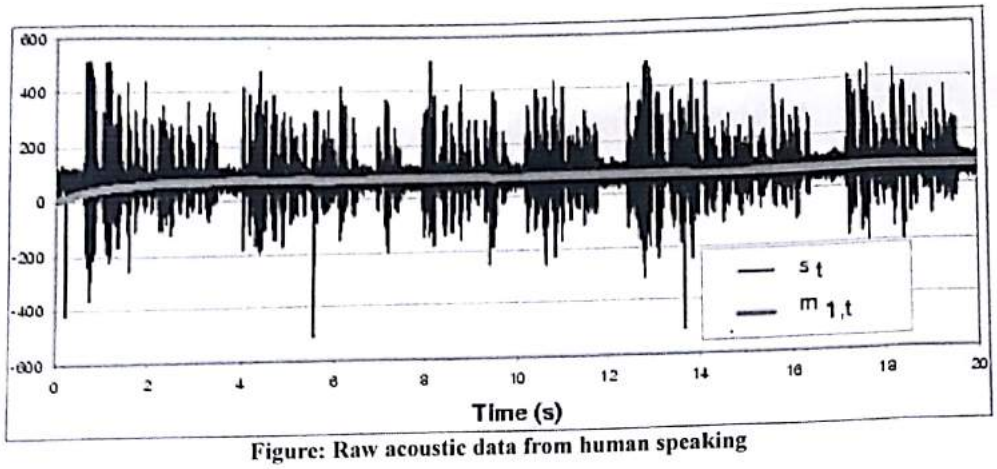

Figure: Raw acoustic data from human speaking

In the RAM of the base mote, a small data structure for each target is maintained. The data structure includes the recent location of the target, the latest timestamp, accumulated sensor values and a pointer to the information of the last report for the target. The base mote chooses the target whose recent location is the closest to the location of the incoming report and decides that the report belongs to the target. If there is no target or the closest distance from the recent location of any target to the location of the reports is greater than a predefined threshold, the report is considered to be from a new target. This threshold needs to be tuned in real-system testing. If it is too large, reports from multiple targets may be categorized into one group. If it is too small, two consecutive reports from a single target may be categorized into two groups. Currently in our system we use a threshold of 60 meters, which shows good performance results in experiments. In order to minimize the number of false positives, a target is reported to the front end interface only if the number of reports for it exceeds a predefined threshold. With this approach, most sporadic false positives can be filtered out.

Once a target accumulates enough reports, the base mote reads its history and applies a linear regression to calculate the velocity of the target, because velocity is one of the most important aspects for moving target tracking, and it is of great interest to the end users. The least square regression approach has been used in many scientific and engineering fields for a long time and is believed to be highly robust against small numbers of outliers. For each direction, the timestamps and the coordinates of the locations of the the most recent reports are used in the regression. The least square algorithm gives the average changing rate of the coordinates over time. This rate serves as the component of velocity along the direction. With the information of both velocity components, we can get the velocity of the target including the knowledge of its moving direction.

4. Protocol Stack for wireless sensor network(wsn)

The protocol stack for WSNs consists of five protocol layers: the physical layer, data link layer, network layer, transport layer, and application layer

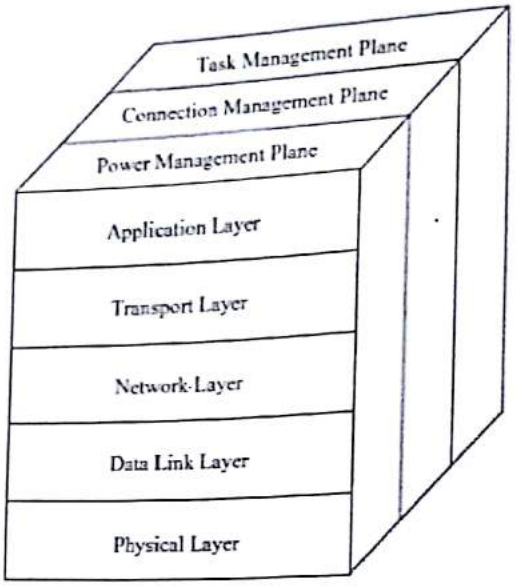

Fig: Protocol stack for sensor networks.

On the other hand, the protocol stack can be divided into a group of management planes across each layer, including power, connection, and task management planes. The power management plane is responsible for managing the power level of a sensor node for sensing, processing, and transmission and reception, which can be implemented by employing effi cient power management mechanisms at different protocol layers.

a. Application Layer

The application layer includes a variety of application - layer protocols that perform various sensor network applications, such as query dissemination, node localization, time synchronization, and network security. For example, the sensor management protocol (SMP) [1] is an application - layer management protocol that provides software operations to perform a variety of tasks, for example, exchanging location - related data, synchronizing sensor nodes, moving sensor nodes, scheduling sensor nodes, and querying the status of sensor nodes. The sensor query and data dissemination protocol (SQDDP) provides user applications with interfaces to issue queries, respond to queries, and collect responses.

b. Transport Layer

In general, the transport layer is responsible for reliable end - to - end data delivery between sensor nodes and the sink(s). Due to the energy, computation, and storage constraints of sensor nodes, traditional transport protocols cannot be

applied directly to sensor networks without modifi cation. For example, the conventional end - to - end retransmission – based error control and the window – based congestion control mechanisms used in the transport control protocol (TCP) cannot be used for sensor networks directly because they are not effi cient in resource utilization.

On the other hand, sensor networks are application specifi c. A sensor network is usually deployed for a specifi c sensing application, for example, habitat monitoring, inventory control, and battlefi eld surveillance. Different applications may have different reliability requirements, which have a big impact on the design of transport - layer protocols.

c. Network Layer

The network layer is responsible for routing the data sensed by source sensor nodes to the data sink(s). In a sensor network, sensor nodes are deployed in a sensing region to observe a phenomenon of interest. The observed phenomenon or data need to be transmitted to the data sink. In general, a source node can transmit the sensed data to the sink either directly via single - hop long - range wireless communication or via multihop short - range wireless communication. However, long - range wireless communication is costly in terms of both energy consumption and implementation complexity for sensor nodes.

d. Data Link Layer

The data link layer is responsible for data stream multiplexing, data frame creation and detection, medium access, and error control in order to provide reliable point - to - point and point - to - multipoint transmissions. One of the most important functions of the data link layer is medium access control (MAC). The primary objective of MAC is to fairly and effi ciently share the shared communication resources or medium among multiple sensor nodes in order to achieve good network performance in terms of energy consumption, network throughput, and delivery latency. However, MAC protocols for traditional wireless networks cannot be applied directly to sensor networks without modifi cation because they do not take into account the unique characteristics of sensor networks, in particular, the energy constraint.

e. Physical Layer

The physical layer is responsible for converting bit streams from the data link layer to signals that are suitable for transmission over the communication medium.For this purpose, it must deal with various related issues, for example, transmission medium and frequency selection, carrier frequency generation, signal modulation and detection, and data encryption. In addition, it must also deal with the design of the underlying hardware, and various electrical and mechanical interfaces.

Unit 3

1. Fundamental MAC Protocol

Medium access control is critical for enabling successful network operation in all shared - medium networks. The primary task of a MAC protocol is to arbitrate access to a shared medium or channel in order to avoid collision and at the same time to fairly and efficiently share the bandwidth resources among multiple nodes.

a. Contention-Based MAC Protocols

In contention - based MAC, all nodes share a common medium and contend for the medium for transmission. Thus, collision may occur during the contention process. To avoid collision, a MAC protocol can be used to arbitrate access to the shared channel through some probabilistic coordination. Both ALOHA (Additive Link On - Line Hawaii System) and CSMA are the most typical examples of contention - based MAC protocols [3] . In pure ALOHA, a node simply transmits whenever it has a packet to send. In the event of a collision, the collided packet is discarded. The sender just waits a random period of time and then transmits the packet again. In slotted ALOHA, time is divided into discrete timeslots. Each node is allocated a timeslot. A node is not allowed to transmit until the beginning of the next timeslot. Pure ALOHA is easy to implement. However, its problem is that the channel effi ciency is only ~ 10% [3] . Compared with pure ALOHA, slotted ALOHA can double the channel effi ciency. However, it requires global time synchronization, which complicates the system implementation.

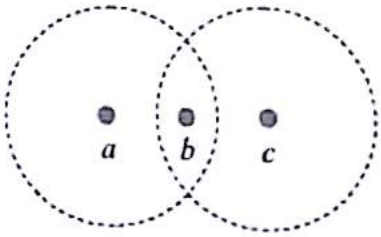

Fig. 3.1 Illustration of the hidden-node problem.

b. Contention-Free MAC Protocols

In contention - free MAC, a shared medium is divided into a number of subchannels in terms of time, frequency, or orthogonal pseudo - noise codes. These subchannels are allocated to individual nodes with each node occupying one subchannel. This allows different nodes to access the shared medium without interfering with each other and thus effectively avoids collision from different nodes.

The most typical examples of contention - free MAC protocols are TDMA,

FDMA, and CDMA [3] . TDMA divides the shared channel into a fi xed number of timeslots and confi gures these timeslots into a frame that repeats periodically. Each node is allocated a timeslot and is allowed to transmit only in the allocated timeslot in each frame. TDMA has been widely used in wireless cellular systems. In a typical cellular system, the base station in each cell allocates timeslots and provides information for time synchronization to all mobile nodes. The mobile nodes communicate only with the base station without direct peer - to - peer communication between each other.

FDMA divides the shared channel into a number of non - overlapping frequency subbands and allocates these subbands to individual nodes. Each node can transmit at any time, but only at a different frequency to avoid interfering with the others. The major advantage of FDMA is its simplicity in implementation. However, it also has some drawbacks.

CDMA divides the shared channel by using orthogonal pseudo - noise codes, rather than timeslots in TDMA and frequency bands in FDMA. All nodes can transmit in the same channel simultaneously, but with different pseudo – noise codes. The major advantage of CDMA is that it does not require strict time synchronization and avoids the channel allocation problem in FDMA. However, it also has some disadvantages. For example, it introduces the energy consumption for coding and decoding. The capacity of a CDMA system in the presence of noise is usually lower than that of a TDMA system.

2. MAC design for WSN

a. Network Characteristics

To better understand WSNs, let us fi rst take a brief look at some conventional wireless networks, for example, wireless cellular networks, mobile ad hoc networks (MANETs), and wireless LANs.

A cellular system is a wireless network consisting of both stationary and mobile nodes. The stationary nodes, or base stations, are connected by wired links, forming a fi xed infrastructure. The number of mobile nodes is much larger than that of stationary nodes. Each base station usually covers a large region with little overlap and serves tens to hundreds of mobile nodes in the region. Each mobile node is only a single - hop away from its closest base station. The primary goal of a cellular system is to provide quality of service and bandwidth effi ciency. The base stations have suffi cient power supply and the mobile users can conveniently replace the batteries in their handsets. Accordingly, energy conservation is only of secondary importance.

In contrast to all the above conventional networks, WSNs have the following unique characteristics:

• A sensor network typically consists of a larger number of sensor nodes densely deployed in a geographical fi eld. The number of sensor nodes can be several orders of magnitude larger than that of conventional wireless networks.

• Sensor nodes are usually powered by battery and thus are limited in power capacity. It is often diffi cult or impossible to change or recharge batteries for these nodes. The lifetime of a sensor network largely depends on the lifetime of its sensor nodes.

• Sensor nodes are often deployed in an ad hoc fashion without careful planning and engineering. Hence, they must be able to organize themselves into a communication network.

• The topology of a sensor network changes more frequently due to both node failure and mobility. Sensor nodes are prone to failures. Most sensor nodes are stationery after deployment. But in some applications, some sensor nodes can also be mobile.

• Sensor nodes have very limited computational capacity and memory.

b. **Objectives of MAC Design**

The basic function of a MAC protocol is to arbitrate access to a shared medium in order to avoid collisions from different nodes. In addition to this basic function, a MAC protocol must also take into account other factors in its design in order to improve network performance and provide good network services for different applications.

- *Energy Effi ciency.*

Energy effi ciency is one of the most important factors that must be considered in MAC design for sensor networks. It refers to the energy consumed per unit of successful communication. Since sensor nodes are usually battery powered and it is often very diffi cult or impossible to change or recharge batteries for sensor nodes.

• *Scalability.* Scalability refers to the ability to accommodate the change in network size. In sensor networks, the number of sensor nodes deployed may be on the order of tens, hundreds, or thousands. A MAC protocol must be scalable to such changes in network size.

• *Adaptability.* Adaptability refers to the ability to accommodate the changes in node density and network topology. In sensor networks, node density can be very high. A node may fail, join, or move, which would result in changes in node density and network topology. A MAC protocol must be adaptive to such changes effi ciently.

• *Channel Utilization.* Channel utilization refers to the bandwidth utilization for effective communication. Due to limited bandwidth, a MAC protocol should make use of the bandwidth as effi ciently as possible.

• *Latency.* Latency refers to the delay from the time a sender has a packet to send until the time the packet is successfully received by the receiver. In sensor networks, the importance of latency depends on different applications. While it is true that latency is not a critical factor for some applications (e.g., data collection for scientifi c exploration), many applications may have stringent latency requirements (e.g., real - time monitoring of bush fi res).

• *Throughput.* Throughput refers to the amount of data successfully transferred from a sender to a receiver in a given time, usually measured in bits or bytes per second. It is affected by many factors, for example, the effi - ciency of collision avoidance, control overhead, channel utilization, and latency. Like latency, the importance of throughput depends on different applications.

• *Fairness.* Fairness refers to the ability of different sensor nodes to equally share a common transmission channel. In some traditional networks, it is important to achieve fairness for each user in order to ensure the quality of service for their applications. In sensor networks, however, all nodes cooperate to accomplish a single common task. What is important is not to achieve per - node fairness, but to ensure the quality of service for the whole task.

C. Energy Effi ciency in MAC Design

Energy efficiency is of primary importance in WSNs. In general, energy consumption occurs in three aspects: sensing, data processing, and data communication, where data communication is a major source of energy consumption.

Thus, sensor nodes can use their processing capability to locally perform simple data processing, instead of sending all raw data to the sink(s) for processing, and then transmit partially processed data to the sink(s) for further processing. On the other hand, an effi cient MAC protocol can improve energy efficiency in data communication and prolong the lifetime of a sensor network. To design an energy - effi cient MAC protocol, it is important to identify the major sources of energy waste in sensor networks from the MAC perspective.

Energy waste comes from four major sources: collision, overhearing, control overhead, and idle listening.

• *Collision.* Collision occurs when two sensor nodes transmit their packets at the same time. As a result, the packets are corrupted and thus have to be discarded. Retransmissions of the packets increase both energy consumption and delivery latency.

• *Overhearing.* Overhearing occurs when a sensor node receives packets that are destined for other nodes. Overhearing such packets results in unnecessary waste of energy and such waste can be very large when traffi c load is heavy and node density is high.

• *Idle Listening.* Idle listening occurs when a sensor node is listening to the radio channel to receive possible data packets while there are actually no data packets sent in the network. In this case, the node will stay in an idle state for a long time, which results in a large amount of energy waste. However, in many MAC protocols, for example, IEEE 802.11 ad hoc mode or CSMA, a node has to listen to the channel to receive possible data packets.

• *Control Overhead.* A MAC protocol requires sending, receiving, and listening to a certain necessary control packets, which also consumes energy not for data communication.

3. S-MAC

The sensor - MAC (S - MAC) protocol proposed by Ye et al. is an energy - effi cient MAC protocol specifi cally designed for WSNs. S - MAC considers a sensor network scenario in which most communication occurs between nodes as peers, rather than to a single base station, and its applications have long idle periods and can tolerate latency on the order of network messaging time. The primary goal of the S - MAC design is to improve energy efficiency while maintaining good scalability and collision avoidance. To achieve this goal, S - MAC tries to reduce energy consumption from all the major sources that cause ineffi cient use of energy. In exchange, it allows some performance degradation in both per - hop fairness and latency. This is implemented by integrating several effective control mechanisms into a contention - based MAC protocol built on top of the IEEE 802.11 standard. These mechanisms include periodic listen and

To reduce idle listening, S - MAC introduces a periodic listen and sleep mechanism to establish a low - duty - cycle operation on each node. With this mechanism, each node is periodically put into a sleep state for some time, and then wakes up and listens to see if it needs to communicate with any other node.

In the sleep state, the radio is completely turned off and a timer is set to awake the node at a later time.

To reduce idle listening, S - MAC introduces a periodic listen and sleep mechanism to establish a low - duty - cycle operation on each node. With this mechanism each node is periodically put into a sleep state for some time, and then wakes u and listens to see if it needs to communicate with any other node. In the sleep state, the radio is completely turned off and a timer is set to awake the node at a later time. A complete cycle of listen and sleep periods is called a frame. Each frame begins with a listen period, during which a node can communicate with the other nodes, followed by a sleep period, during which a node sleeps if it has no data to send or receive, or remains awake if it has data to send or receive.

In S - MAC, all nodes are free to choose their own listen and sleep schedules. To reduce control overhead, however, neighboring nodes coordinate their sleep schedules and try to adopt the same schedules to listen and sleep, rather than randomly sleep on their own.

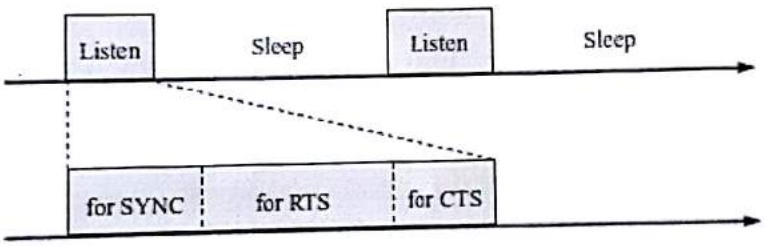

Fig. 3.2 Periodic listen and sleep in S-MAC

4. ***DS-MAC.*** DS - MAC is an S - MAC protocol with a dynamic duty cycle proposed by Lin et al. [16] , which aims to achieve a good tradeoff between energy consumption and latency without incurring much overhead. In DS - MAC, each sensor node assumes all functionalities defi ned in S - MAC and each receiver node keeps track of its own energy consumption level and average latency. To achieve the intended tradeoff, each node attempts to dynamically adjust its sleep – wakeup cycle time based on the current energy consumption level and the average latency it has experienced. The average latency is used as an approximate estimation of the current traffi c condition and an indicative parameter for a receiver node.

With DS - MAC, each node uses the SYNC packets to set up and maintain clock synchronization as done similarly in S - MAC. Unlike S - MAC, which adopts a constant duty cycle, DS – MAC adopts a common initial basic

duty cycle at all sensor nodes. If a receiver node fi nds that the latency becomes intolerable, it will double the original duty cycle by reducing the sleeping period accordingly without changing the listening period. As a result, a node with an increased duty cycle can get more chances to receive packets from other senders instead of blocking them while sleeping. Therefore, DS - MAC alleviates the high - latency problem with S - MAC under high - traffi c load while still keeping high energy effi ciency under low traffi c load.

To implement DS - MAC, some additional protocol overhead needs to be introduced, including a " duty cycle " fi eld and a " delay " fi eld in each SYNC packet. Compared with the S - MAC implementation, each sensor node also needs to maintain its own average latency and energy consumption level, which requires additional storage overhead and processing overhead. However, all these overheads are negligible and can actually be compensated by the reduced queuing cost due to the decreased latency.

5. ***MS-MAC***

MS - MAC is an adaptive mobility - aware MAC protocol proposed by Pham and Jha to address the mobility issue in mobile sensor applications like smart patient assistance and rare animal monitoring. In such mobile sensor applications, each sensor node could be highly mobile and the level of mobility may vary signifi cantly during different periods of a day. Before MS - MAC, most MAC protocols proposed for WSNs only consider stationery sensor nodes, which may largely degrade the network performance if directly applied to mobile scenarios. To improve the network performance in mobile scenarios, a MAC protocol must be mobility aware and able to adapt to different levels of mobility. To this end, MS - MAC adopts the design of S - MAC and extends the protocol to support mobile sensor nodes. For a stationery scenario, MS - MAC operates similar to S - MAC in order to conserve energy. For a highly mobile scenario, it switches to an operating mode similar to IEEE 802.11. The protocol uses any change in the received signal levels of periodical SYNC messages as an indication of mobility and if necessary triggers a mobility handling mechanism, which dynamically adjusts the frequency of mobility handling actions based on the presence of mobile nodes and their moving speeds. With such a mobility estimating and handling mechanism, MS - MAC is highly energy efficient for stationery scenarios while also maintaining a certain level of network performance in scenarios with mobile sensor nodes.

6. *Traffic-Adaptive Medium Access*

The Traffic-Adaptive Medium Access (TRAMA) protocol (Rajendran *et al.* 2003) isa contention-free MAC protocol that aims to increase the network throughput and energy-efficiency, compared to traditional TDMA and contention-based solutions. It uses a distributed election scheme based on information about the traffic at each node to determine when nodes are allowed to transmit. This helps to avoid assigning slots to nodes with no traffic to send (leading to increased throughput) and allows nodes to determine when they can become idle and do not have to listen to the channel (increased energy efficiency).

TRAMA assumes a time-slotted channel, where time is divided into periodic randomaccess intervals (signaling slots) and scheduled-access intervals (transmission slots). During random-access intervals, the Neighbor Protocol (NP) is used to propagate one-hop neighbour information among neighboring nodes, allowing them to obtain consistent two-hop topology information. During the random-access interval, nodes join a network by transmitting during a randomly selected slot. The packets transmitted during these slots are used to gather neighbourhood information by carrying a set of added and deleted neighbors. In case no changes have occurred, these packets serve as "keep-alive" beacons. By collecting such updates, a node knows the one-hop neighbors of its own one-hop neighbors, thereby obtaining information about its two-hop neighborhood.

A second protocol, called the Schedule Exchange Protocol (SEP), is used to establish and broadcast actual schedules, that is, allocations of slots to a node. Each node computes a duration SCHEDULE_INTERVAL, which represents the number of slots for which the node can announce its schedule to its neighbors. This duration depends on the rate at which the node's applications can produce packets. At time t , the node then computes the number of slots within [t, t+ SCHEDULE_INTERVAL] for which it has the highest priority among its two-hop neighbors. The node announces the selected slots and the intended receivers using a *schedule packet*. The last slot in this schedule is used to announce the next schedule for the next interval. For example, if a node's SCHEDULE_INTERVAL is 100 slots and the current time (slot number) is 1000, a possible slot selection for interval [1000, 1100] for this node could be 1011, 1021, 1049, 1050, and 1093. During slot 1093, the node broadcasts its new schedule for interval [1093, 1193].

The list of intended receivers in the schedule packet is implemented as a bitmap, whose length is equal to the number of one-hop neighbors. Each bit in the bitmap corresponds to one particular receiver ordered by its identity. Since every node knows the topology within its two-hop neighborhood, it can determine the receiver address based on the bitmap and its list of neighbors.

Slot selection is based on the node's priority at time t using a pseudo-random hash of the concatenation of the node's identity i and t:
$prio(i, t) = hash(i \oplus t)$.

In summary, TRAMA reduces the probability of collisions and increases the sleep time (and energy savings) compared to CSMA-based protocols. Unlike standard TDMA approaches, TRAMA divides time into random-access and scheduled-access intervals. During the random-access intervals, nodes are awake to either transmit or receive topology information, that is, the length of the random-access interval (relative to the scheduled-access interval) affects the overall duty cycle and achievable energy savings of a node.

7. Self organizing MAC

Unit-4

1. Data dissemination and gathering

The way that data and queries are forwarded between the base station and the location where the target phenomena are observed is an important aspect and a basic feature of WSNs. A simple approach to accomplishing this task is for each sensor node to exchange data directly with the base station. A single-hop-based approach, however, is costly, as nodes that are farther away from the base station may deplete their energy reserves quickly, thereby severely limiting the lifetime of the network. This is the case particularly where the wireless sensors are deployed to cover a large geographical region or where the wireless sensors are mobile and may move away from the base station.

To address the shortcomings of the single-hop approach, data exchange between the sensors and the base stations is usually carried out using multihop packet transmission over short communication radius. Such an approach leads to significant energy savings and reduces considerably communication interference between sensor nodes competing to access the channel, particularly in highly dense WSNs. Data forwarding between the sensors where data are collected and the sinks where data are made available is illustrated in Figure 6.2. In response to queries issued by the sinks or when specific events occur within the area monitored, data collected by the sensors are transmitted to the base station using multihop paths. It is worth noting that depending on the nature of the application, sensor nodes can aggregate data correlated on their way to the base station.

In a multihop WSN, intermediate nodes must participate in forwarding data packets between the source and the destination. Determining which set of intermediate nodes is to be selected to form a data-forwarding path between the source and the destination is the principal task of the routing algorithm. In general, routing in large-scale networks is inherently a difficult problem whose solution must address multiple challenging design requirements, including correctness, stability, and optimality with respect to various performance metrics. The intrinsic properties of WSNs, combined with severe energy and bandwidth constraints, bring about additional challenges that must be addressed to satisfy the traffic requirements of the application supported, while extending the lifetime of the network.

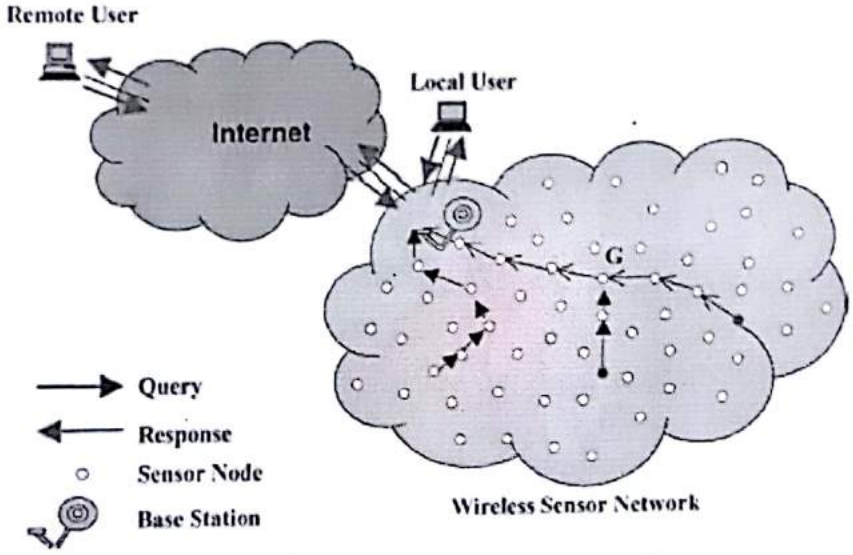

Figure 6.2 Multihop data and query forwarding.

2. ROUTING CHALLENGES AND DESIGN ISSUES IN WIRELESS SENSOR NETWORKS

Although WSNs share many commonalities with wired and ad hoc networks, they also exhibit a number of unique characteristics which set them apart from existing networks. These unique characteristics bring to sharp focus new routing design requirements that go beyond those typically encountered in wired and wirelessad hoc networks.

a. Network Scale and Time-Varying Characteristics

Due to the large number of conceivable sensor-based applications, the densities of the WSNs may vary widely, ranging from very sparse to very dense. Furthermore, in many applications, the sensor nodes, in some cases numbering in the hundreds if not thousands, are deployed in an ad hoc and often unsupervised manner over wide coverage areas. In these networks, the behavior

of sensor nodes is dynamic and highly adaptive, as the need to self-organize and conserve energy forces sensor nodes to adjust their behavior constantly in response to their current level of activity or the lack thereof. Furthermore, sensor nodes may be required to adjust their behavior in response to the erratic and unpredictable behavior of wireless connections caused by high noise levels and radio-frequency interference, to prevent severe performance degradation of the application supported.

b. Resource Constraints

Sensor nodes are designed with minimal complexity for large-scale deployment at a reduced cost. Energy is a key concern in WSNs, which must achieve a long lifetime while operating on limited battery reserves. Multihop packet transmission over wireless networks is a major source of power consumption. Reducing energy consumption can be achieved by dynamically controlling the duty cycle of the wireless sensors. The energy management problem, however, becomes especially challenging in many mission-critical sensor applications. The requirements of these applications are such that a predetermined level of sensing and communication performance constraints must be maintained simultaneously. Therefore, a question arises as to how to design scalable routing algorithms that can operate efficiently for a wide range of performance constraints and design requirements. The development of these protocols is fundamental to the future of WSNs.

c. Sensor Applications Data Models

The data model describes the flow of information between the sensor nodes and the data sink. These models are highly dependent on the nature of the application in terms of how data are requested and used. Several data models have been proposed to address the data-gathering needs and interaction requirements of a variety of sensor applications .A class of sensor applications requires data collection models that are based on periodic sampling or are driven by the occurrence of specific events. In other applications, data can be captured and stored, possibly processed and aggregated by a sensor node, before they are forwarded to the data sink. Yet a third class of sensor applications requires bidirectional data models in which two-way interaction between sensors and data sinks is required .

The need to support a variety of data models increases the complexity of the routing design problem. Optimizing the routing protocol for an application's specific data requirements while supporting a variety of data models and delivering the highest performance in scalability, reliability, responsiveness, and power efficiencybecomes a design and engineering problem of enormous magnitude.

3. ROUTING STRATEGIES IN WIRELESS SENSOR NETWORKS

The WSN routing problem presents a very difficult challenge that can be posed as a classic trade-off between responsiveness and efficiency. Routing algorithms for ad hoc networks can be classified according to the manner in which information is acquired and maintained and the manner in which this information is used to compute paths based on the acquired information. Three different strategies can be identified: proactive, reactive, and hybrid.

The proactive strategy, also referred to as table driven, relies on periodic dissemination of routing information to maintain consistent and accurate routing tables across all nodes of the network. The structure of the network can be either flat or hierarchical. Flat proactive routing strategies have the potential to compute optimal paths. The overhead required to compute these paths may be prohibitive in a dynamically changing environment. Hierarchical routing is better suited to meet the routing demands of large ad hoc networks.

Reactive routing strategies establish routes to a limited set of destinations on demand. These strategies do not typically maintain global information across all nodes of the network. They must therefore, rely on a dynamic route search to establish paths between a source and a destination. This typically involves floodinga route discovery query, with the replies traveling back along the reverse path. The reactive routing strategies vary in the way they control the flooding process to reduce communication overhead and the way in which routes are computed and reestablished when failure occurs.

Hybrid strategies rely on the existence of network structure to achieve stability and scalability in large networks. In these strategies the network is organized into mutually adjacent clusters, which are maintained dynamically as nodes join and leave their assigned clusters. Clustering provides a structure that can be leveraged to limit the scope of the routing algorithm reaction to changes in the network environment. A hybrid routing strategy can be adopted whereby proactive routing is used within a cluster and reactive routing is used across clusters. The main challenge is to reduce the overhead required to maintain the clusters (its type is in wsn technology pdf read while reading this question)

4. Flooding and it's variants

Flooding is a common technique frequently used for path discovery and information dissemination in wired and wireless ad hoc networks. The routing strategy is simple and does not rely on costly network topology maintenance and complex route discovery algorithms. Flooding uses a reactive approach

whereby each node receiving a data or control packet sends the packet to all its neighbors. After transmission , a packet follows all possible paths . Unless the network is disconnected, the packet will eventually reach its destination. Furthermore, as the network topology changes, the packet transmitted follows the new routes. Figure 6.3 illustrates the concept of flooding in data communications network. As shown in the figure, flooding in its simplest form may cause packets to be replicated indefinitely by network nodes.

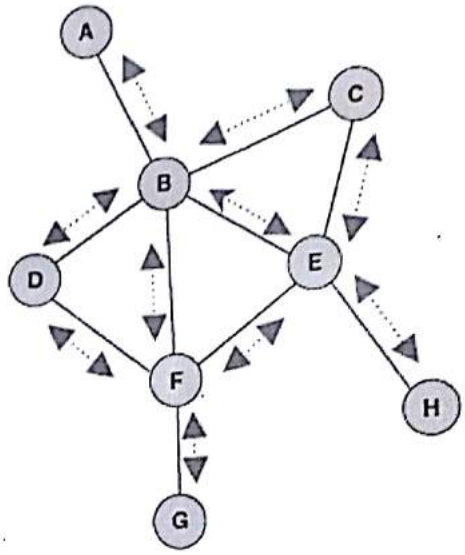

Figure 6.3 Flooding in data communications networks.

To prevent a packet from circulating indefinitely in the network, a hop count field is usually included in the packet. Initially, the hop count is set to approximately the diameter of the network. As the packet travels across the network, the hop count is decremented by one for each hop that it traverses. When the hop count reaches zero, the packet is simply discarded. A similar effect can be achieved using a time-to-live field, which records the number of time units that a packet is allowed to live within the network. At the expiration of this time, the packet is no longer forwarded. Flooding can be further enhanced by identifying data packets uniquely, forcing each network node to drop all the packets that it has already forwarded. Such a strategy requires maintaining at least a recent history of the traffic, to keep track of which data packets have already been forwarded.

Despite the simplicity of its forwarding rule and the relatively low-cost maintenance that it requires, flooding suffers several deficiencies when used in WSNs. The first drawback of flooding is its susceptibility to traffic implosion, as shown in Figure 6.4. This undesirable effect is caused by duplicate control or data packets being sent repeatedly to the same node. The second drawback of flooding is the overlap problem to which it gives rise, as depicted in Figure 6.5.

Overlapping occurs when two nodes covering the same region send packets containing similar information to the same node. The third and most severe drawback of flooding is resource blindness. The simple forwarding rule that flooding uses to route packets does not take into consideration the energy constraints of the sensor nodes. As such, the node's energy may deplete rapidly, reducing considerably the lifetime of the network.

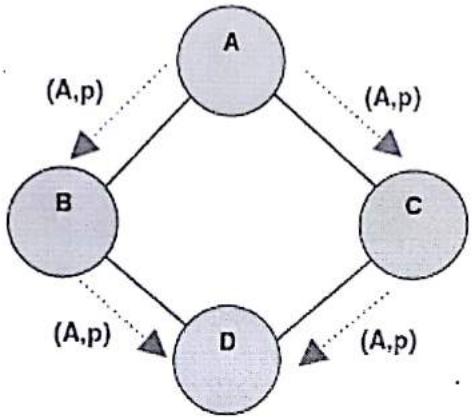

Figure 6.4 Flooding traffic implosion problem.

To address the shortcomings of flooding, a derivative approach, referred to as gossiping, has been proposed . Similar to flooding, gossiping uses a simple forwarding rule and does not require costly topology maintenance or complex route discovery algorithms. Contrary to flooding, where a data packet is broadcast to all neighbors, gossiping requires that each node sends the incoming packet to a randomly selected neighbor. Upon receiving the packet, the neighbor selected randomly chooses one of its own neighbors and forwards the packet to the neighbour chosen. This process continues iteratively until the packet reaches its intended destination or the maximum hop count is exceeded. Gossiping avoids the implosion problem by limiting the number of packets that each node sends to its neighbor to one copy. The latency that a packet suffers on its way to the destination may be excessive, particularly in a large network. This is caused primarily by the random nature of the protocol, which, in essence, explores one path at a time.

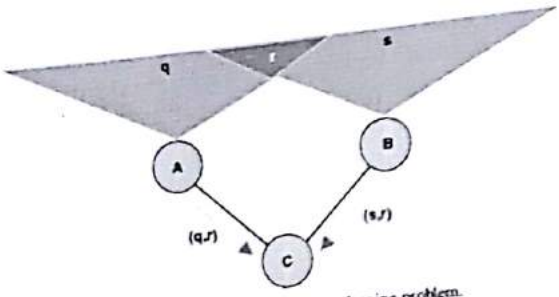

Figure 6.5 Flooding traffic overlapping problem.

5. Low energy adaptive clustering

Low-energy adaptive clustering hierarchy (LEACH) is a routing algorithm designed to collect and deliver data to the data sink, typically a base station . The main objectives of LEACH are:

- Extension of the network lifetime
- Reduced energy consumption by each network sensor node
- Use of data aggregation to reduce the number of communication messages

To achieve these objectives, LEACH adopts a hierarchical approach to organize the network into a set of clusters. Each cluster is managed by a selected cluster head. The cluster head assumes the responsibility to carry out multiple tasks. The first task consists of periodic collection of data from the members of the cluster. Upon gathering the data, the cluster head aggregates it in an effort to remove redundancy among correlated values [6.19,6.20]. The second main task of a cluster head is to transmit the aggregated data directly to the base station. The transmission of the aggregated data is achieved over a single hop. The network model used by LEACH is depicted in Figure 6.9. The third main task of the cluster head is to create a TDMA-based schedule whereby each node of the cluster is assigned a time slot that it can use for transmission. The cluster head advertises the schedule to its cluster
members through broadcasting. To reduce the likelihood of collisions among sensors within and outside the cluster, LEACH nodes use a code-division multiple access–based scheme for communication.

The basic operations of LEACH are organized in two distinct phases. These phases are illustrated in Figure 6.10. The first phase, the setup phase, consists of two steps, cluster-head selection and cluster formation. The second phase, the steady-state phase, focuses on data collection, aggregation, and delivery to the base station. The duration of the setup is assumed to be relatively shorter than the steady-state phase to minimize the protocol overhead.

At the beginning of the setup phase, a round of cluster-head selection starts. The

cluster-head selection process ensures that this role rotates among sensor nodes, thereby distributing energy consumption evenly across all network nodes. To determineif it is its turn to become a cluster head, a node, n, generates a random number,v, between 0 and 1 and compares it to the cluster-head selection threshold, Tðnþ. The node becomes a cluster head if its generated value, v, is less than Tðnþ. The cluster-head selection threshold is designed to ensure with high probability that a predetermined fraction of nodes, P, is elected cluster heads at each round. Further, the threshold ensures that nodes which served in the last 1=P rounds are not selected in the current round.
To meet these requirements, the threshold T(n) of a competing node n can be expressed as follows:

$$T(n) = \begin{cases} 0 & \text{if } n \notin G \\ \dfrac{P}{1 - P(r \bmod (1/P))} & \forall n \in G \end{cases}$$

The variable G represents the set of nodes that have not been selected to become cluster heads in the last 1=P rounds, and r denotes the current round. The predefined

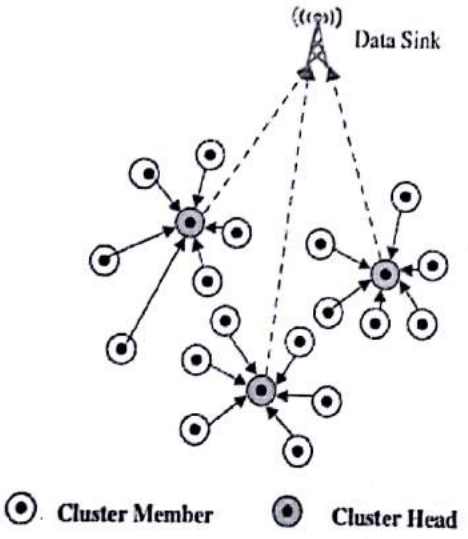

Figure 6.9 LEACH network model.

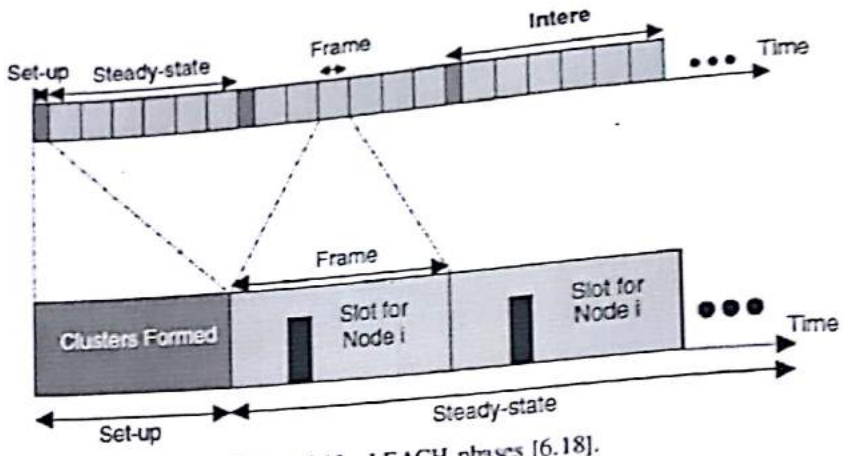

Figure 6.10 LEACH phases [6.18].

parameter, P, represents the cluster-head probability. It is clear that if a node has served as a cluster head in the last 1=P rounds, it will not be elected in this round.

At the completion of the cluster-head selection process, every node that was selected to become a cluster head advertises its new role to the rest of the network. Upon receiving the cluster-head advertisements, each remaining node selects a cluster to join. The selection criteria may be based on the received signal strength, among other factors. The nodes then inform their selected cluster head of their desire to become a member of the cluster.

Upon cluster formation, each cluster head creates and distributes the TDMA schedule, which specifies the time slots allocated for each member of the cluster. Each cluster head also selects a CDMA code, which is then distributed to all members of its cluster. The code is selected carefully so as to reduce intercluster interference. The completion of the setup phase signals the beginning of the steady-state phase. During this phase, nodes collect information and use their allocated slots to transmit to the cluster head the data collected. This data collection is performed periodically.

Simulation results show that LEACH achieves significant energy savings. These savings depend primarily on the data aggregation ratio achieved by the cluster heads. Despite these benefits, however, LEACH suffers several shortcomings. The assumption that all nodes can reach the base station in one hop may not be realistic, as capabilities and energy reserves of the nodes may vary over time from one node to another. Furthermore, the length of the steady-state period is critical to achieving the energy reduction necessary to offset the overhead caused by the cluster selection process. A short steady-state period increases the protocol's overhead, whereas a long period may lead to cluster head energy depletion. Several algorithms have been proposed to address these

shortcomings. The extended LEACH (XLEACH) protocol takes into consideration the node's energy level in the cluster-head selection process. The resulting threshold cluster-head selection,
T(n), used by n to determine if it will be a cluster head in the current round is defined as

$$T(n) = \frac{P}{1 - P(r \bmod (1/P))} \left[\frac{E_{n,\text{current}}}{E_{n,\max}} + \left(r_{n,s} \operatorname{div} \frac{1}{P} \right) \left(1 - \frac{E_{n,\text{current}}}{E_{n,\max}} \right) \right]$$

In this equation, En;current is the current energy, and En;max is the initial energy of the sensor node. The variable rn;s is the number of consecutive rounds in which a node has not been a cluster head. When the value of rn;s approaches 1=P, the threshold T(n) is reset to the value it had before the inclusion of the remaining energy onto the threshold equation. Additionally, rn;s is set to 0 when a node becomes a cluster head.

LEACH exhibits several properties which enable the protocol to reduce energy consumption. Energy requirement in LEACH is distributed across all sensor nodes, as they assume the cluster head role in a round-robin fashion based on their residual energy. LEACH is a completely distributed algorithm, requiring no control information from the base station. The cluster management is achieved locally, which obliterates the need for global network knowledge. Furthermore, data aggregation by the cluster also contributes greatly to energy saving, as nodes are no longer required to send their information directly to the sink. It has been shown using simulation that LEACH outperforms conventional routing protocols, including direct transmission and multihop routing, minimum-transmission-energy routing, and static clustering–based routing algorithms.

6. Geographical Routing

The main objective of geographical routing is to use location information to formulate an efficient route search toward the destination. Geographical routing is very suitable to sensor networks, where data aggregation is a useful technique to minimize the number of transmissions toward the base station by eliminating redundancy among packets from different sources.

In addition to its compatibility with data-centric applications, geographical routing requires low computation and communication overhead. In traditional routing approaches such as the one used in distributed shortest-path routing protocols for wired networks, knowledge of the entire network topology, or a summary thereof, may be required for a router to compute the shortest path to each destination . Furthermore, to maintain correct paths to all destinations, routers are called upon to update the state describing the current topology in a

periodic fashion and when link failure occurs. The need to update the topology state constantly may lead to substantial overhead, proportional to the product of the number of routers and the rate of topological changes in the network.

Geographical routing, on the other hand, does not require maintaining a ''heavy'' state at the routers to keep track of the current state of the topology. It requires only the propagation of single-hop topology information, such as the position of the ''best'' neighbor to make correct forwarding decisions. The self-describing nature of geographical routing, combined with its localized approach to decision, obliterates the need for maintaining internal data structures such as routing tables. Consequently, the control overhead is reduced substantially, thereby enhancing its scalability in large networks. These attributes make geographical routing a feasible solution for routing .in resource-constrained sensor networks.

Routing Strategies The objective of geographical routing is to use location information to formulate a more efficient routing strategy that does not require Flooding request packets throughout a network. To achieve this goal, a data packet is sent to nodes located within a designated forwarding region. In this scheme, also referred to as geocasting, only nodes that lie within the designated forwarding zone are allowed to forward the data packet [6.23,6.24]. The forwarding region can be statically defined by the source node, or constructed dynamically by intermediate nodes to exclude nodes that may cause a detour when forwarding the data packet. If a node does not have information regarding the destination, the route search can begin as a fully directed broadcast. Intermediate nodes, with better knowledge of the destination, may limit the forwarding zone in order to direct traffic toward the destination. The idea of limiting the scope of packet propagation to a designated region is commensurate with the data-centric property of sensor networks, in which the interest in the data content, rather than the sensor, provides the data. The efficacy of the strategy depends largely on the way the designated forwarding is defined and updated as data travel toward the destination. It also depends on the connectivity of the nodes within a designated zone.

A second strategy used in geographical routing, referred to as position-based routing, requires a node to know only the location information of its direct neighbours [6.25,6.26]. A greedy forwarding mechanism is then used whereby each node forwards a packet to the neighboring node that is ''closest'' to the destination. Several metrics have been proposed to define the concept of closeness, including the Euclidean distance to the destination, the projected distance to the destination on the straight line joining the current node and the destination, and the deviation from the straight direction toward the destination.

Position-based routing protocols have the potential to reduce control overhead and reduce energy, as flooding for node discovery and state propagation are localized to within a single hop. The efficiency of the scheme, however, depends on the network density, the accurate localization of nodes, and more important, on the forwarding rule used to move data traffic toward the destination. In the following section, various forwarding rules commonly used in position-based routing are described. Basic techniques used to overcome the lack of position information and obstacles are described.

Forwarding Approaches An important aspect of geographical routing is the rule used to forward traffic toward its final destination. In position-based routing, each node decides on the next hop based on its own position, the position of its neighbors, and the destination node. The quality of the decision clearly depends on the extent of the node's knowledge of the global topology [6.27]. Local knowledge of the topology may lead to suboptimal paths, as depicted in Figure 6.15, where the node currently holding the packet makes a forwarding decision global knowledge of based solely on local topology knowledge. Finding the optimal path requires the topology. The overhead that global knowledge of the topology entails, however, is prohibitive in resource-constrained WSNs. To overcome this problem, various forwarding strategies have been proposed .

The greedy routing scheme selects among its neighbors the one that is closest to the destination. In Figure 6,16, the node currently holding the message, node MH, selects node GRS as the next hop to forward the message. It is worth noting that the selection process used in this scheme considers only the set of nodes that are closer to the destination than the current message holder. If such a set is empty, the scheme fails to progress forward.

In the most-forward-within-R strategy (MFR), where R represents the transmission range, a node transmits its packet to the most forward among its neighbours toward the destination. Based on this approach, the next hop selected by MH to forward the packet is node MFR. This greedy approach is myopic and does not necessarily minimize the remaining distance to the destination.

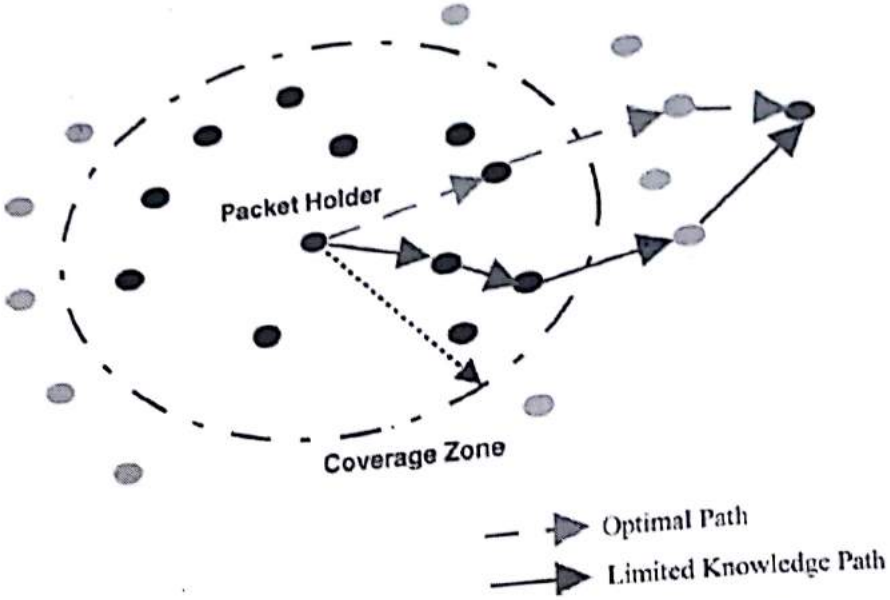

Figure 6.15 Localized and globalized forwarding decision.

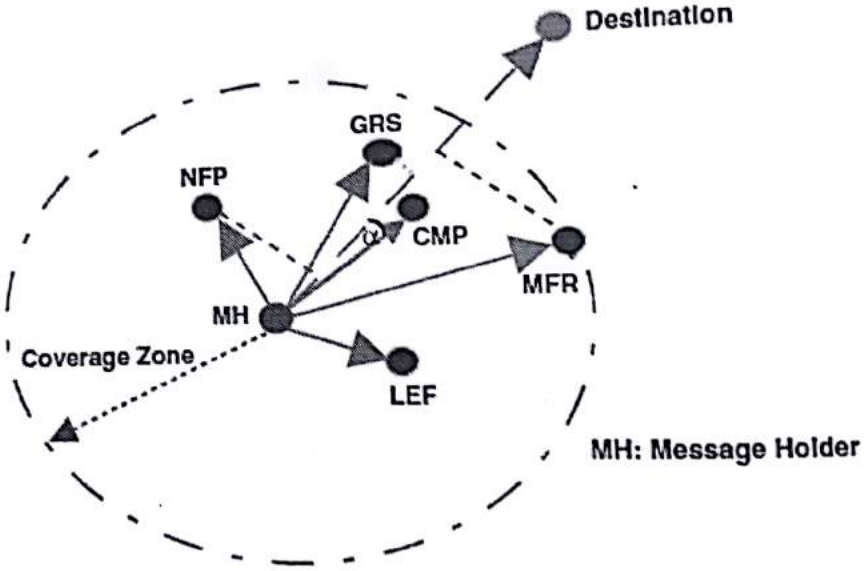

Figure 6.16 Geographical routing forwarding strategies.

Unit 5

1. Traditional transport protocol

This section introduces the principles of traditional transport protocols and discuss the disadvantages of TCP and UDP

a. Principles of Traditional Transport Protocols

The architecture of a computer communication network is often stratifi ed into several different layers, including the physical, the data - link, the network, the transport, and the upper layers, for example, application layer. Each lower layer serves as a service provider transparently providing some services to its upper layer, the service customer, through so - called service access points (SAPs). Forexample, the data - link layer aims to provide a reliable link to the network layer.The network layer provides addressing service and routing service to the transport layer, which in turn provides end - to - end message transportation service to the upper layers. In this layered network model, the lower three layers sit in each intermediate network node or communication entity; however, the transport and the upper layers generally exist only in end points or hosts, and belong to end - to - end protocols.

The transport layer sits on the network layer. It utilizes the services provided by the network layer to enable end - to - end message transportation, where messages are fragmented to chains of segments at senders and reassembled into original messages at receivers, and does not concern with the underlying path carrying segments. A protocol at the transport layer is called a transport protocol, for example, TCP [7] , UDP [8] , and the Stream Control Transmission Protocol (SCTP) [9] . Both TCP and UDP are standard protocols that have been widely deployed in the Internet for many years. SCTP is a new transport protocol with the aim to support reliable signaling transmission.

Transport protocols can be generally classifi ed into two types: connection oriented and connectionless. A connectionless protocol does not need to establish a connection before data is transferred and has only one phase: *data transmission* . In contrast, a connection - oriented protocol has three phases for each transmission process [10] : *connection establishment* , *data transmission* , and *disconnection* . Moreover, a transport protocol can be *responsive* (e.g., TCP) or *unresponsive* (e.g., UDP). A *responsive* protocol can adjust the source sending rate (e.g., increase or decrease), while an *unresponsive* protocol does not change. A transport protocol, especially a connection - oriented protocol, usually provides the following functions to the upper layers:

• *Orderly Transmission.* Since multiple paths may exist between a source and a destination, a packet that is sent earlier may arrive at the destination later. This phenomenon is called *disordered transmission* . A transport protocol

may need to provide sequential transmission service for real - time applications. The common approach is to piggyback a *sequence number* in the header of each transmitted segment. In this way, the destination can sort the received segments based on the sequence number carried in each segment.

• *Flow Control and Congestion Control.* The hosts often have different characteristics, for example, the capacity of communication and computation. If the source transmits segments with a higher rate than the destination can receive, or in other words the source sending rate exceeds the bottleneck link bandwidth on the path between the source and the destination, congestion will occur and thus incur segment loss. Therefore, the transport layer still needs to provide fl ow control and congestion control service to coordinate the sending rate from the source to the destination.

• *Loss Recovery.* Congestion in a network leads to packet loss because a node has fi nite memory. Although the data - link layer can recover data loss caused by bit errors, it is unable to recover data loss caused by buffer overfl ow. For this reason, a transport protocol should support loss recovery so as to provide a loss - free virtual tunnel to the upper - layer applications, especially those loss - sensitive applications, for example, File Transfer Protocol (FTP).

• *QoS.* For real - time applications, for example, voice over IP (VoIP) that are delay critical and need to sustain a certain bandwidth, a transport protocol should provide low delay and high throughput under the constraints. For this purpose, the transport protocol can incorporate QoS design into fl ow control and congestion control.

b. Disadvantages of TCP and UDP

As two popular transport protocols, TCP and UDP have been broadly deployed in the Internet and wired networks. However, neither of them is a good choice for WSNs. In what follows, we respectively discuss their disadvantages when used in WSNs:

• TCP is a connection - oriented protocol. Before data transmission, there is a three - way handshake interactive process. If and only if after a TCP connection has been established between a TCP sender and a TCP receiver, the TCP sender can begin to transmit data. In WSNs, the sensory data for event - based applications is only several bytes or so (a value of an interest). The three - way handshake process will be a big overhead compared to the small volume of data. Also, since a wireless link is error prone, the time to

set up a TCP connection can be much longer than that in the Internet. Therefore, the data may become outdated after the TCP connection has been established.

• TCP assumes that segment loss is resulted from congestion and triggers window - based fl ow control and congestion control once it detects segment loss. This can incur that TCP unwisely reduces the sending rate under WSNs even if there is no congestion, and further leads to low throughput, especially under a multihop wireless environment. Therefore, it is hard for sensor nodes, especially those far away from the sink, to obtain enough bandwidth to support those applications that require continual data transmissions.

• The control congestion in TCP is end - to - end. This approach usually has a tardy response when congestion occurs, and will thus result in lots of segment dropping. The segment dropping wastes valuable energy and implies low energy effi ciency.

• TCP uses end - to - end acknowledgment (ACK) and retransmission to guarantee reliability. The end - to - end approach not only takes longer time to recover lost segments, but also consumes much energy and reduces network lifetime.

• In WSNs, the sensor nodes may have different hops and different round - trip time (RTT) from the data sink. TCP in such an environment may cause unfairness and make the sensor nodes near the sink get more opportunities to transmit data, and therefore deplete their energy fi rst. In this case, the whole network may become disjointed if no other mechanism is introduced.

• UDP is a connectionless transport protocol. But it is also unsuitable for WSNs considering that: (a) there is no fl ow and congestion control mechanism in UDP. If UDP is used for WSNs, it will cause lots of datagram dropping when congestion occurs. At this point, UDP is at least not energy effi cient for WSNs; (b) UDP contains neither ACK mechanism, nor any reliability mechanism. The datagram loss can only be recovered by lower MAC protocols or upper layers, including the application layer.

2. TRANSPORT PROTOCOL DESIGN FOR WIRELESS SENSOR NETWORKS

A transport protocol runs over the network layer. It enables end - to - end message transmission, where a message may be fragmented into several segments at the transmitter and reassembled at the receiver. This protocol provides the following functions: orderly transmission, fl ow and congestion control, loss recovery, and possibly QoS (e.g., timing and fairness) guarantee.

i. Performance Metrics

In a WSN, a transport protocol should provide end - to - end reliability and end - to -end QoS in an energy - effi cient manner. The performance of a transport protocol can be evaluated using different metrics, for example, energy effi ciency, reliability, QoS (e.g., packet loss ratio or packet delivery latency), and fairness.

Energy Effi ciency. A sensor node usually has limited energy. For this reason, it is most important for a transport protocol to keep high energy effi ciency in order to prolong the network lifetime. Due to bit errors and/or congestion, packet loss is common in a WSN. For loss - sensitive applications, packet loss leads to retransmission and inevitably consumes additional energy. Therefore, packet loss is a primary factor that affects energy effi -ciency at the transport layer. If we defi ne *retransmission distance* as the hop number from the node that requests retransmission to the node that retransmits lost packets, different retransmission mechanisms may have different retransmission distances. A mechanism with a longer retransmission distance consumes more energy. Therefore, retransmission distance is the second factor that affects energy effi ciency. The third factor that affects energy effi ciency is the use of control messages. A transport layer may use control messages to perform congestion control and loss recovery.

Reliability. For different applications, different levels of reliability may be required. There are several defi nitions of reliability in the literature . According to the sensitivity of an application to packet loss, two types of reliability can be classifi ed: *packet reliability* and *event reliability* . *Packet reliability* means that the application is very loss sensitive and requires the successful transmission of each packet. One example of such applications is downstream code distribution or queries. *Event reliability* stands for the requirement of loss - tolerant applications that allow certain packet loss. For example, the sensor nodes with digital camera can be used to send images to the sink. Since images are somewhat loss tolerant, the sink does not need to correctly receive every packet, but only a certain percentage of packets.

ii. **Congestion Control**

There are mainly two reasons that result in congestion in a WSN. The fi rst is the packet arrival rate exceeding the packet service rate. This is more likely to occur at the sensor nodes closer to the sink because they usually carry more combined upstream traffi c. The second reason is contention, interference, and the bit error rate on a link, which can result in congestion on the link.

a. ***Congestion Detection.*** In TCP, congestion is observed or inferred at the end nodes based on a timeout or redundant acknowledgment. In a WSN, however, it is preferred to use proactive mechanisms. A common mechanism is to use queue length, packet service time, or the ratio of packet service time over packet interarrival time at the intermediate nodes. If a network uses a CSMA like MAC protocol, channel loading can be measured and used as an indication of congestion. Therefore, measurement can be used as a means for detecting congestion.

b. ***Congestion Notifi cation.*** After detecting congestion, a transport protocol needs to propagate the congestion information from the congested node to its upstream nodes or the source nodes that contribute to congestion. The information can be transmitted using, for example, a single binary bit, called congestion notifi cation (CN) bit in Refs. [13,17,18] , or more information, for example, allowable data rate as in Ref. [16] , or the congestion degree .

The approaches to disseminate congestion information can be categorized into *explicit congestion notifi cation* and *implicit congestion notifi cation* . The *explicit congestion notifi cation* uses special control messages to notify the involved sensor nodes of congestion information. For example, in Ref. [18] , an intermediate sensor node will broadcast a suppression message upstream toward the source when it perceives congestion. The node receiving the suppression message will continue to relay the message toward the source unless the suppression message has traversed for a certain hops, called *depth of congestion.*

c. ***Congestion Mitigation and Avoidance.*** There are two general approaches to mitigate and avoid congestion: *network resource management* and *traffi c control* . The fi rst approach tries to increase network resources (e.g., bandwidth) to mitigate the congestion when congestion occurs. In a wireless network, power control and multiple radio interfaces can be used to increase bandwidth and mitigate congestion. For example, the virtual sinks in Siphon have two radio interfaces: one primary low - power mote radio with smaller bandwidth and another long - range radio with larger bandwidth. When congestion occurs, the long - range radio is used as a shortcut or " siphon " to

mitigate the congestion. With this approach, it is necessary to guarantee precise and exact network resource adjustment in order to avoid overprovided resources or underprovided resources. However, this is a hard task in wireless environments. Unlike the approaches based on network resource management, traffi c control implies controlling congestion through adjusting the traffi c rate at source nodes or intermediates nodes. This approach is helpful to saving network resources, and is more feasible and efficient when exact adjustment of network resources becomes difficult. Most existing congestion control protocols belong to this type

iv. Loss Recovery

In wireless environments, both congestion and bit errors can cause packet loss, which would degrade end - to - end reliability and QoS, and further decrease energy effi ciency. Other factors that result in packet loss include node malfunction, incorrect or outdated routing information, and energy depletion. In order to address this problem, one can increase the source sending rate or introduce retransmission - based loss recovery. The former approach, which is also used in Event - to - Sink Reliable Transport (ESRT) [13] , is effective for guaranteeing event reliability for event - driven applications that require no packet reliability; however, this approach is not energy effi cient compared to loss recovery. Loss recovery is more active and energy effi cient, and can be performed at both the link layer and the transport layer. At the link layer, loss recovery is performed on a hop - by – hop basis, while at the transport it is usually done on an end - to - end basis. In what follows, we introduce a loss recovery approach that consists of two phases: loss detection and notifi cation, and retransmission recovery.

a. ***Loss Detection and Notifi cation.*** Since packet loss can be far more common in WSNs than in wired networks, a loss detection mechanism has to be carefully designed. A common mechanism is to include a *sequence number* in each packet header. The continuity of *sequence numbers* can be used to detect packet loss. Loss detection and notifi cation can be performed either *end - to – end* or *hop - by - hop* . In the end - to - end approach, the end points (destination or source) are responsible for loss detection and notifi cation as in TCP. In the hop - by – hop approach, intermediate nodes detect and notify packet loss.

b. ***Retransmission Recovery.*** Retransmission of lost or damaged packets can also be performed either end - to - end or hop - by - hop. In the end - to - end retransmission, the source performs the retransmission. In the hop - by – hop retransmission, an intermediate node that intercepts loss notifi cation searches its local buffer. If it fi nds a copy of the lost packet in the buffer, it

retransmits it.Otherwise, it relays the loss information upstream to other intermediate nodes.

v. Design Guidelines

In order to design an effi cient transport protocol, several factors must be taken into consideration, including the network topology, diversity of applications, traffi c characteristics, and resource constraints. The two most signifi cant constraints in WSNs are energy and fairness among different geographically deployed sensor nodes. A transport protocol should provide high energy effi ciency and flexible reliability, as well as QoS in terms of throughout, packet - loss rate and end - to - end delay if necessary.

Therefore, transport protocols for WSNs should have components including congestion control and loss recovery because these have a direct impact on energy effi ciency, reliability, and application QoS. There are generally two approaches to perform this task. The fi rst approach is to design separate protocols or algorithms, respectively, for congestion control and loss recovery. Most existing protocols use this approach and address congestion control or reliable transport separately. With this separate and usually modular design, applications that need reliability can invoke only a loss recovery algorithm, or invoke a congestion control algorithm if they need to control congestion.

3. Message authentication code

In order to deal with false packets, authentication is indispensable to ensure the origin of received packets. *Message authentication code* (MAC) is a tool to solve the problem. It can also be called MIC because it ensures packet integrity as well.

To compute a MAC, a symmetric key shared between the sender and the receiver is required. For a packet payload M, the sender concatenates it with the shared key K and then computes a MAC as $C = H(M \| K)$, where $H(\cdot)$ is a collision - resistant hash function [14] and " $\|$ " is the concatenating operator. The packet including payload M and the MAC C is sent to the receiver. The receiver recomputes a MAC C' with the payload M and the shared key K and then checks whether $C' = C$ holds. If the equation holds, the payload M is authenticated and not modifi ed because only the sender knows the shared key. Otherwise, the packet is either modifi ed or injected.

In TinySec , another type of frame Auth is defi ned in addition to AE. Both Auth and AE frames carry a MAC so that the link - layer authentication is supported between neighboring nodes.

4. Signature

Signature is an asymmetric key technique, which is widely used in authentication. A sender node keeps its private key Ks in secret while publishing its public key Kp . In order to authenticate a plaintext M to the receiver, the sender uses its private key Ks to sign M into a signature $S = S (M , Ks)$ and then transmits the signature S , as well as the plaintext M to the receiver. Because Ks is secret, only the sender can generate the signature. Because Kp is publicly well known, any receiver can verify the signature S by inputting the signature S , the plaintext M , and the public key Kp into a verifi cation algorithm M to compute $V (S , M , Kp)$. If the output is TRUE, the plaintext M is authenticated, and otherwise not.

5. Authenticating Public Key

The reason that the MiM attack is possible is that the authenticity of the public key cannot be assured. Therefore, authenticating public keys in asymmetric key systems is a very critical problem.

The conventional solution to the public key authentication is to rely on a *public key infrastructure* (PKI). In the PKI, there is a *certifi cate authority* (CA), which is trusted by all the members using the PKI. The public key of the CA is accepted by all the member nodes as an authenticated one in default. The CA signs the public key of each member node and issues a certifi cate including the public key and the corresponding signature to the member node. When two nodes need to communicate, one of them sends its public key certifi cate to the other node that can verify the authenticity of the public key in the certifi cate with the well - known public key of the CA. In this way, the two nodes can authenticate each other.

The PKI discussed above can be illustrated as a two - level tree with the CA as the root and all the member nodes as leafs. In reality, a PKI can be described as a multilevel tree. Each non - leaf node acts as a local CA and manages its children CAs at lower levels.

Public key certificates have been widely used in the Internet and other wireless networks, for example, *wireless local area networks* (WLANs). However, there is one obstacle to using public key certificates in WSNs. Most asymmetric key algorithms, for example, Diffie – Hellman [4] and RSA [5] are very expensive on resource - constrained sensor platforms. Therefore, how to effi ciently perform asymmetric key algorithms becomes a very important and popular issue in WSNs.

One approach is to use specifi c parameters that can speed asymmetric key

algorithms without compromising security too much. TinyPK [16] is an example of using RSA based public key certifi cates in WSNs. In particular, an external user needs to acquire a certifi cate from the network administrator, who acts as the CA. The certifi cate is verifi ed by sensor nodes so that the user may be authorized to access the network to collect data or issue commands. In order to simplify the certifi cate verifi cation in sensor nodes, the CA ' s RSA public key is chosen as 3, while an ordinary value of a RSA public key for a satisfying security is usually hundreds of bits long. Meanwhile, researchers are looking for new algorithms that are more effi cient than traditional asymmetric key algorithms. A more promising technique is the *elliptic curve cryptography* (ECC) .

6. Broadcast and Multicast Authentication

Broadcast/multicast is a common mechanism to disseminate information from a source node to a group of destination nodes. As in unicast, each packet in broadcast/multicast should also be authenticated. This can be done by using the symmetric key technique or the asymmetric key technique.

A simple symmetric key technique is to confi gure all the nodes in a broadcast/multicast group with a shared group key, which is an extension of the shared pairwise key in unicast authentication. A source node simply attaches each outgoing packets with a MAC computed with the group key and all the destination nodes verify the MAC with the group key. This method assumes that all the group members are trustful. This assumption may fail if an attacker is able to compromise any member node. A compromised member node can impersonate the source node and spread false information because he also knows the group key.

In order to solve the problem of internal attackers, a special symmetric key technique called *one-way hash chain* (OHC) can be used. An OHC is a sequence of numbers $\{K_0, K_1, \ldots, K_n\}$, such that $K_{j-1} = H(K_j)$, $\forall j \in \{1, 2, \ldots, n\}$, where the hash function $H(\cdot)$ satisfies two properties:

1. Given x, it is easy to computer $y = H(x)$.
2. Given y, it is computationally infeasible to compute x such that $y = H(x)$.

The first number of the OHC, K_0, is securely sent to all the receiving nodes as a commitment. When the source node needs to broadcast/multicast a packet in the kth round, it includes K_k in the packet. Then every member node can authenticate K_k by verifying whether

$$H^k(K_k) = H^{k-1}(K_{k-1}) = \ldots = H(K_1) = K_0$$

holds. However, K_k cannot be directly used to authentjcate packets in the kth round. The reason is that if an attacker has the ability to capture a packet, he can

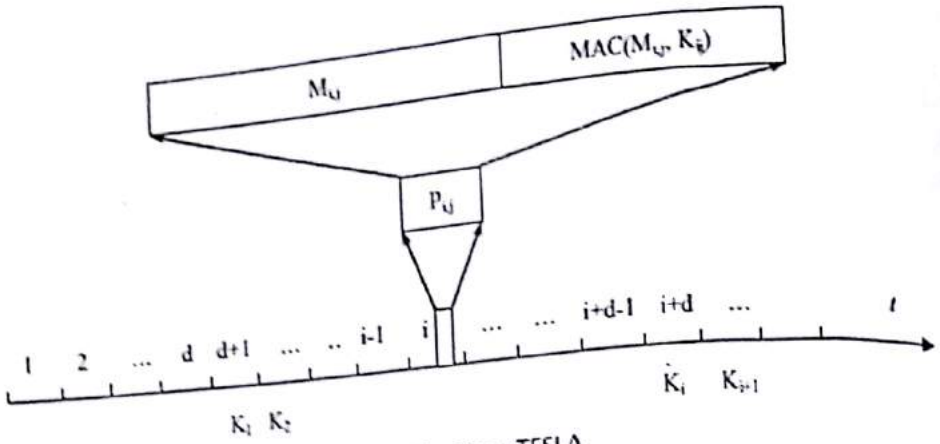

Fig. 12.2 The μTESLA.

extract the key attached to it, generate a false packet, and send it to a node in the same round before the true packet arrives at that node.

By using the OHC technique, μ TESLA in SPINS [24] simulates the asymmetry of the asymmetric key technique and provides authenticated broadcast services from the base station. In μTESLA, the fi rst key of the OHC is sent to all the receiving nodes as a commitment in advance, as shown in Fig. 12.2 . A broadcast packet in the k th time slot carries a MAC generated by using the k th key K_k of the OHC. Every node does not know the k th key when it receives the packet. In the ($k + d$)th time slot, the base station discloses the k th key K_k, which is used by every node to authenticate the cached packet. This introduced asymmetry by delayed key release can effi ciently prevent malicious nodes from impersonating the source node.

In Refs. [27,28], a novel broadcast/multicast authentication protocol called *multicast authentication based on batch signature* (MABS) uses an efficient asymmetric cryptographic primitive called *batch signature*, which supports the authentication of any number of packets simultaneously. In particular, a sender generates a signature for each outgoing packet with its private key. When a receiver collects n packets $p_i = \{m_i, s_i\}$, $i = 1, \ldots, n$, where m_i is the data payload, s_i is the corresponding signature, and n can be any positive integer, the receiver can input them into an algorithm, *BatchVerify*$(p_1, p_2, \ldots, p_n) \in \{True, False\}$. If the output is *True*, then the n packets are authentic. Otherwise, they are not authentic.

In order to support authenticity and efficiency, the algorithm *BatchVerify*() should satisfy the following properties: (1) given a batch of packets that have been signed by the sender, *BatchVerify*() outputs *True*, (2) given a batch of packets including some unauthentic packets, the probability that *BatchVerify*() outputs *True* is very low, and (3) the computational complexity of *BatchVerify*() is comparable to that of verifying one signature and is increased gradually when the batch size n increases.

By using batch signature, each receiver can achieve the computational effi -

ciency comparable to conventional block - based schemes in the sense that a batch of packets can be authenticated simultaneously through one batch signature verifi cation operation. The MABS protocol also eliminates the correlation among packets, and thus is perfectly resilient to packet loss in the sense that no matter how many packets are lost, the rest can also be verifi ed by receivers. In addition, using per packet signature instead of per block signature eliminates the authentication latency at the sender and/or receivers. Each receiver can verify the authenticity of all the received packets in its buffer anytime.

Unit 6

I. Traditional network management models

i.Simple Network Management Protocol

The simple network management protocol (SNMP) for managing networks is in broad use today. It includes three components: a network management system (NMS), managed elements, and agents. NMS is a set of applications that monitor and/or control managed elements. It can request management information (or attributes) from the agent and present the results to NM users in the form of figures or tables. It can also set attributes within the agent. The managed elements are the network devices that are managed. SNMP agents run on each managed element. The managed elements collect and store management information in the MIB and provide access through SNMP to the MIBs. Examples of managed elements include routers, switches, servers, and hosts. SNMP agents are management software modules that reside on managed elements. Agents collect and store the state of the managed elements and translate this information into a form compatible with SNMP MIB. Exchanges of network management information are through messages called protocol data units (PDUs). These are sent to nodes and contain variables that have both attributes and values. The SNMP defines five types of messages or PDUs: Two deal with the reading terminal, another two handle terminal configuration, and the fifth is Trap, used to monitor events in the managed elements. Each PDU contains both attributes and values. NM information can be exchanged through the PDUs in order to monitor the managed elements.

An advantage of SNMP is its simplicity and wide deployment. However, it consumes considerable bandwidth since it often gets only one piece of management information at a time: GetRequest (GetNextRequest) and GetResponse. Although in SNMP version 3 it can obtain more information by a pair of PDUs such as (GetBulkRequest and GetResponse), due to the usually large number of managed elements, large bandwidth consumption still exists.

The other disadvantage of SNMP is that it only manages network elements; it does not support network-level management.

ii.Telecom Operation Map

The telecom operation map (TOM), proposed by TeleManagement Forum [9.15], is based on the service management and network management process models. TOM presents a model for telecommunications management for network and service management and a view of "operations." The idea behind TOM is to introduce processes comprising operations and their automation. There are three vertical layers for service management: network and systems management, service development and operations, and customer care process. Horizontally, the service management is divided in service fulfillment, service assurance, and service billing. TOM only provides a framework for service management.

Neither SNMP nor TOM is designed particularly for wireless sensor networks. However, one can utilize the simplicity of SNMP and the layered framework of TOM to design effective and efficient network management architecture for wireless sensor networks as well.

2. NETWORK MANAGEMENT DESIGN ISSUES

WSN is a special type of wireless network, possibly with ad hoc structure and probably with limited resources. Due to these WSN constraints, networking protocols, the application model, middleware, and sensor node operating systems should be designed very carefully. Network management for WSNs is required to use those limited resources effectively and efficiently. Network management is much more important for WSNs than for traditional networks for the following reasons:

1. In order to deploy an adaptive and resource-efficient algorithm in WSNs, the current resource level needs to be gathered through network management. For example, the power availability should be known before switching a sensor node from active (or sleep) mode to sleep (or active) mode. Most traditional networks do not have these requirements.

2. Most WSN applications need to know the coverage area so that they ensure that the entire space is being monitored. Topology management can be used in case an uncovered area is detected. Generally, there are three approaches to increasing the coverage area: (1) increase the node's radio power, (2) increase the density of deployment of senor nodes, and (3) move the sensor nodes around to achieve equal distribution.

3. Nodes in WSNs are usually arranged in an ad hoc manner. The parameters of

this ad hoc network are obtained by the network management system.

4. Collaboration and cooperation between sensor nodes are required to optimize system performance. Network management is an effective tool to provide the platform required for this purpose.

So far, very little attention has been paid to the management of WSNs. An issue is whether in the meanwhile, any of the existing network management solutions (e.g., SNMP [9.14], TOM [9.15]) can be used for WSNs. SNMP is often used to manage network elements such as switches and routers. It uses GetResponse and GetResponse PDUs to collect information from network elements. In SNMP, a local management agent should run in each managed element. The local agent is a static and passive agent that receives commands from a manager and returns the corresponding response. It can also issue Trap messages to the manager when the managed element encounters a preconfigured event. Agents in different network elements are independent, and there is not collaboration among them. TOM is a new operation and management model that provides a layered architecture for management and administration. Each layer has a different management function and set of managed objects. TOM can be used to manage most tasks, from the underlying physical network element to the entire network, as well as the services provided. However, SNMP is just a simple protocol that only manages network elements. Given that WSNs are data centric, resource constrained, and ad hoc, SNMP and TOM, which were designed for traditional networks, may not provide the right tool.

The issue of management architecture for WSNs should also be considered carefully. A network management platform consists of three major components: manager, agent, and MIB. The manager is used to manage and control the entire network and works as an interface to other systems. The agent is located in managed elements. MIB is an object-oriented structured tree that informs the manager and agent about the organization of management information. A standardized MIB guarantees that the management products from different vendors interconnect. The manager receives management information and commands the managed elements using a SNMP-like method or mobile-agent-based entities. Sometimes a network management system would include several distributed managers, each of which manages part of the entire network. The method of accessing management information and the placement of the manager or agent usually determines the management architecture. The agent-based method can save bandwidth since it can report only final management information. Although WSNs have a centralized data collecting point (sink), they are more like distributed networks. As a result, agent-based hybrid management architectures might be more suitable for WSNs.

Network management functions should therefore consider all the special features of WSNs. Some of these considerations follow:

_ Management solutions should be energy efficient, using as little wireless bandwidth as possible since communication is highly energy demanding.

_ Management solutions should be scalable. This is especially important since it future WSNs may consist of tens to thousands of nodes.

_ Management solutions should be simple and practical since WSNs are resource-constrained distributed systems.

_ MIB for WSNs should contain a general information model for sensor nodes, features of WSNs, and WSN applications.

_ Management solutions for WSNs should provide a general interface to the applications since applications can perform better when able to access management information.

_ Management solutions should be implementable as middleware.

3. EXAMPLE OF MANAGEMENT ARCHITECTURE: MANNA

Several references, notably [9.1–9.13], have extensive discussions and some results on WSN network management. Specifically, [9.1] has an initial discussion of the topics, management architecture is discussed in [9.2] and [9.3], monitoring management in [9.5], resource management is discussed in [9.6] and [9.11], secure management in [9.7] and [9.13], topology management in [9.8], and data stream management in [9.10]. An optimization problem for monitoring management formulated in [9.5] provides the monitoring regions given that the battery and energy consumption rate for each sensor are known beforehand. In topology management scheme called sparse topology and energy management (STEM) proposed in [9.8], the nodes only need to be awake when there is data to forward. Golab and Ozsu [9.10] present an overview of data stream management.

MANNA is a management architecture for WSNs proposed by Ruiz et al. [9.2,9.3]. The architecture considers three management dimensions: function areas, management levels, and WSN functionalities (see Figure 9.1). The management function areas contain five types of traditional management functions similar to SNMP: fault, configuration, performance, security, and accounting management. But configuration management has a notably more important role in MANNA, where all other functions depend on it. The management levels in MANNA are similar to those in TOM: network element, network element management, network management, service management, and business management. A number of other functions are proposed by MANNA: configuration, maintenance, sensing, processing, and communication. With the aim of promoting productivity and integrating the functions of configuration,

operation, administration, and maintenance of all elements and services in a WSN. MANNA architecture includes three architectural elements: functional, physical, and informational architectures. The functional architecture provides functions executed in the management entities (manager, agent, and MIB) and the location scheme for managers and agents. The physical architecture is where functional architecture is implemented. MANNA uses a lightweight protocol as a communication interface between management entities. The information architecture element provides an object-oriented model for mapping manageable resources and supporting object classes. MANNA defines the following managed object classes: (1) network (information on network behavior and features such as data delivery model, network structure, and mobility), (2) managed elements (such as sensor nodes), (3) equipment (the physical components of sensor nodes), (4) system (information on operating system), (5) environment (the environment the WSN is running), (6) phenomenon, and (7) connection.

MANNA lists several common management functions for WSNs: environment monitoring functions, a coverage area supervision function, a topology map discovery function, an energy-level discovery function, an energy map generation function, and several others. It also provides a dynamic MIB model for WSNs: a sensing coverage area map, a communication coverage area map, a WSN behaviour model, a node dependence model, network topology, residual energy, and so on. In MANNA, the management functions have the lowest granularity and can be combined into management services.

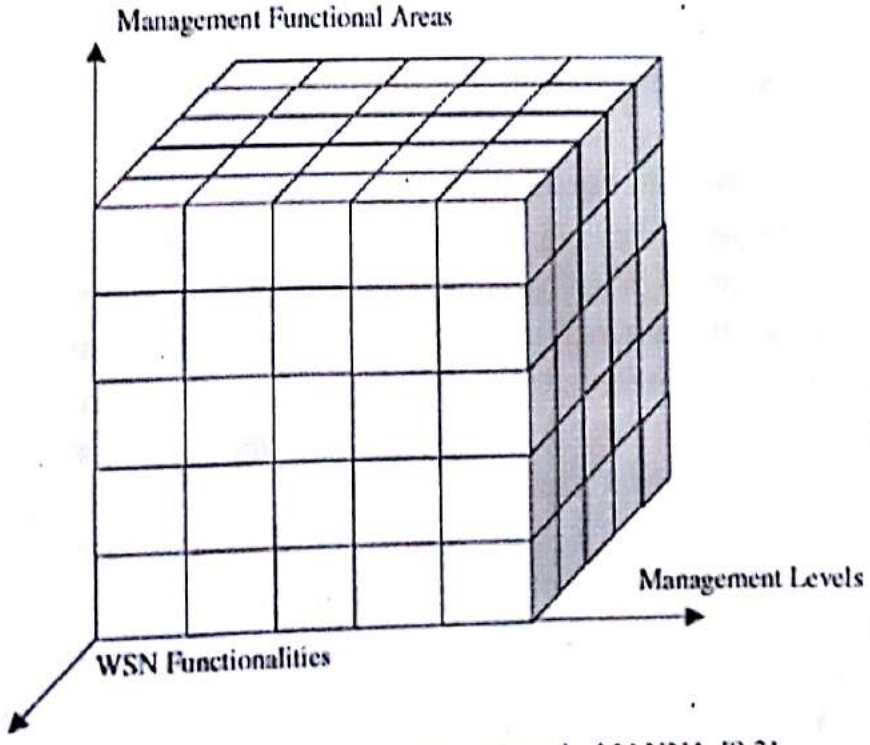

Figure 9.1 Management functions in MANNA [9.3].

4. Operating system design issues

Traditional operating systems [10, 17, 19, 20] are system software, including programs that manage computing resources, control peripheral devices, and provide software abstraction to the application software. Traditional OS functions are therefore to manage processes, memory, CPU time, file system, and devices. This is often implemented in a modular and layered fashion, including a lower layer of kernels and a higher layer of system libraries. Traditional OSs are not suitable for wireless sensor networks because WSNs have constrained resources and diverse data centric applications, in addition to a variable topology. WSNs need a new type of operating system, considering their special characteristics. There are several issues to consider when designing operating systems for wireless sensor networks.

The first issue is process management and scheduling. The traditional OS provides process protection by allocating a separate memory space (stack) for each process. Each process maintains data and information in its own space. But this approach usually causes multiple data copying and context switching between processes. This is obviously not energy efficient for WSNs. For some real-time applications in WSNs, a real-time scheduler such as earliest deadline first (EDF) or its variantsmay be a good choice, but the number of processes should be confined since thatwould determine the time complexity of the EDF scheduler.

The second issue is memory management. Memory is often allocated exclusively for each process task in traditional operating systems, which is helpful for protection and security of the tasks. Since sensor nodes have small memory, another approach, sharing, can reduce memory requirements.

The third issue is the kernel model. The event-driven and finite state machine (FSM) models have been used to design microkernels for WSNs. The event-driven model may serve WSNs well because they look like event-driven systems. An event may comprise receiving a packet, transmitting a packet, detection of an event of interest, alarms about energy depletion of a sensor node, and so on. The FSM-based model is convenient to realize concurrency, reactivity, and synchronization.

The fourth issue is the application program interface (API). Sensor nodes need to provide modular and general APIs for their applications. The APIs should enable applications access the underlying hardware.

The fifth issue is code upgrade and reprogramming. Since the behavior of sensor nodes and their algorithms may need to be adjusted either for their functionality or for energy conservation, the operating system should be able to

reprogram and upgrade. Finally, because sensor nodes generally have no external disk, the operating system for WSNs cannot have a file system. These issues should be considered carefully in the design of WSN OSs and to meet their constrained resources, network behavior, and data-centric application requirements.

Sensor operating systems (SOS) should embody the following functions, bearing in mind the limited resource of sensor nodes:

1. Should be compact and small in size since the sensor nodes have very small memory. The sensor nodes often have memories of only tens or hundreds of kilobytes.

2. Should provide real-time support, since there are real-time applications, especially when actuators are involved. The information received may become outdated rather quickly. Therefore, information should be collected and reported as quickly as possible.

3. Should provide efficient resource management mechanisms in order to allocate microprocessor time and limited memory. The CPU time and limited memory must be scheduled and allocated for processes carefully to guarantee fairness (or priority if required).

4. Should support reliable and efficient code distribution since the functionality performed by the sensor nodes may need to be changed after deployment. The code distribution must keep WSNs running normally and use as little wireless bandwidth as possible.

5. Should support power management, which helps to extend the system lifetime and improve its performance. For example, the operating system may schedule the process to sleep when the system is idle, and to wake up with the advent of an incoming event or an interrupt from the hardware.

6. Should provide a generic programming interface up to sensor middleware or application software. This may allow access and control of hardware directly, to optimize system performance.

5. TinyOS

The design of TinyOS [10.1,10.3] allows application software to access hardware directly when required. TinyOS is a tiny microthreaded OS that attempts to address two issues: how to guarantee concurrent data flows among hardware devices, and how to provide modularized components with little processing and storage overhead. These issues are important since TinyOS is

required to manage hardware capabilities and resources effectively while supporting concurrent operation in an efficient manner. TinyOS uses an event-based model to support high levels of concurrent application in a very small amount of memory. Compared with a stack-based threaded approach, which would require that stack space be reserved for each execution context, and because the switching rate of execution context is slower than in an event-based approach, TinyOS achieves higher throughput. It can rapidly create tasks associated with an event, with no blocking or polling. When CPU is idle, the process is maintained in a sleep state to conserve energy.

TinyOS includes a tiny scheduler and a set of components. The scheduler schedules operation of those components. Each component consists of four parts: command handlers, event handlers, an encapsulated fixed-size frame, and a group of tasks [10.3]. Commands and tasks are executed in the context of the frame and operate on its state. Each component will declare its commands and events to enable modularity and easy interaction with other components. The current task scheduler in TinyOS is a simple FIFO mechanism whose scheduling data structure is very small, but it is power efficient since it allows a processor to sleep when the task queue is empty and while the peripheral devices are still running. The frame is fixed in size and is assigned statically. It specifies the memory requirements of a component at compile time and removes the overhead from dynamic assignment [10.1]. Commands are nonblocking requests made to the low-level components. Therefore, commands do not have to wait a long time to be executed. A command provides feedback by returning status indicating whether it was successful (e.g., in the case of buffer overrun or of timeout). A command often stores request parameters into its frame and conditionally assigns a task for later execution. The occurrence of a hardware event will invoke event handlers. An event handler can store information in its frame, assign tasks, and issue high-level events or call low-level commands. Both commands and events can be used to perform a small and usually fixed amount of work as well as to preempt tasks. Tasks are a major part of components. Like events, tasks can call low-level commands, issue high-level events, and assign other tasks. Through groups of tasks, TinyOS can realize arbitrary computation in an event-based model. The design of components makes it easy to connect various components in the form of function calls.

This WNS operating system defines three type of components: hardware abstractions, synthetic hardware, and high-level software components. Hardware abstraction components are the lowest-level components. They are actually the mapping of physical hardware such as I/O devices, a radio transceiver, and sensors. Each component is mapped to a certain hardware abstraction. Synthetic hardware components are used to map the behavior of

advanced hardware and often sit on the hardware abstraction components. TinyOS designs a hardware abstract component called the radio-frequency module (RFM) for the radio transceiver, and a synthetic hardware component called radio byte, which handles data into or out of the underlying RFM.

An evaluation of TinyOS shows that it achieves the following performance gains or advantages:

- It requires very little code and a small amount of data.
- Events are propagated quickly and the rate of posting a task and switching the corresponding context is very high.
- It enjoys efficient modularity.

6. Mate

Mate [10.2] is designed to work on the top of TinyOS as one of its components. It is a byte-code interpreter that aims to make TinyOS accessible to nonexpert programmers and to enable quick and efficient programming of an entire sensor network. Mate also provides an execution environment, which is helpful for the UC–Berkeley mote (see Chapter 7 for an overview of the mote) since in this system there is no hardware protection mechanism. In Mate, a program code is made up of capsules. Each capsule has 24 instructions, and the length of each instruction is 1 byte. The capsules contain type and version information, which makes code injection easy. Mate capsules can deploy themselves into the network. Mate implements a beaconless (BLESS) ad hoc routing protocol as well as the ability to implement new routing protocols. A sensor node that receives a newer version of a capsule installs it. Through hop-by-hop code injection, Mate can program the entire network. Capsules are classified into four categories: message send, message receive, timer, and subroutine. An event can trigger Mate to run. It can be used not only as a virtual machine platform for application development, but also as a tool to manage and control the entire sensor network.

7. MagnetOS

MagnetOS [10.4] is a distributed adaptive operating system designed specifically for application adaptation and energy conservation. Other operation systems do not provide a network-wide adaptation mechanism or policies for application to effectively utilize the underlying node resources. The burden of creating adaptation mechanisms (if any) is on the application itself. This approach is usually not energy efficient. The goals of MagnetOS are (1) to adapt to the underlying resource and its changes in a stable manner, (2) to be efficient with respect to energy conservation, (3) to provide general abstraction for the applications, and (4) to be scalable for large networks.

MagnetOS is a single system image (SSI) or a single unified Java virtual machine that includes static and dynamic components. The static components rewrite the application in byte-code level and add necessary instructions on the semantics of the original applications. The dynamic components are used for application monitoring, object creation, invocation, and migration. SSI abstraction provides more freedom in object placement and simplifies application development. MagnetOS provides an interface to programmers for explicit object placement and override of the automatic object placement decisions. This OS also provides two online power-aware algorithms (NetPull and NetCenter) for use in moving application components within the entire network so as to reduce energy consumption and extend network lifetime. Netpull works hop by hop at the physical layer, and NetCenter runs multihop at the network level. The difference between traditional ad hoc routing and NetPull (NetCenter) is that the communication endpoints in ad hoc routing are fixed, whereas NetPull tries to move the communication endpoints in order to conserve energy [10.4].

advanced hardware and often sit on the hardware abstraction components. TinyOS designs a hardware abstract component called the radio-frequency module (RFM) for the radio transceiver, and a synthetic hardware component called radio byte, which handles data into or out of the underlying RFM.

An evaluation of TinyOS shows that it achieves the following performance gains or advantages:

_ It requires very little code and a small amount of data.

_ Events are propagated quickly and the rate of posting a task and switching the corresponding context is very high.

_ It enjoys efficient modularity.

6. Mate

Mate [10.2] is designed to work on the top of TinyOS as one of its components. It is a byte-code interpreter that aims to make TinyOS accessible to nonexpert programmers and to enable quick and efficient programming of an entire sensor network. Mate also provides an execution environment, which is helpful for the UC–Berkeley mote (see Chapter 7 for an overview of the mote) since in this system there is no hardware protection mechanism. In Mate, a program code is made up of capsules. Each capsule has 24 instructions, and the length of each instruction is 1 byte. The capsules contain type and version information, which makes code injection easy. Mate capsules can deploy themselves into the network. Mate implements a beaconless (BLESS) ad hoc routing protocol as well as the ability to implement new routing protocols. A sensor node that receives a newer version of a capsule installs it. Through hop-by-hop code injection, Mate can program the entire network. Capsules are classified into four categories: message send, message receive, timer, and subroutine. An event can trigger Mate to run. It can be used not only as a virtual machine platform for application development, but also as a tool to manage and control the entire sensor network.

7. MagnetOS

MagnetOS [10.4] is a distributed adaptive operating system designed specifically for application adaptation and energy conservation. Other operation systems do not provide a network-wide adaptation mechanism or policies for application to effectively utilize the underlying node resources. The burden of creating adaptation mechanisms (if any) is on the application itself. This approach is usually not energy efficient. The goals of MagnetOS are (1) to adapt to the underlying resource and its changes in a stable manner, (2) to be efficient with respect to energy conservation, (3) to provide general abstraction for the applications, and (4) to be scalable for large networks.

MagnetOS is a single system image (SSI) or a single unified Java virtual machine that includes static and dynamic components. The static components rewrite the application in byte-code level and add necessary instructions on the semantics of the original applications. The dynamic components are used for application monitoring, object creation, invocation, and migration. SSI abstraction provides more freedom in object placement and simplifies application development. MagnetOS provides an interface to programmers for explicit object placement and override of the automatic object placement decisions. This OS also provides two online power-aware algorithms (NetPull and NetCenter) for use in moving application components within the entire network so as to reduce energy consumption and extend network lifetime. Netpull works hop by hop at the physical layer, and NetCenter runs multihop at the network level. The difference between traditional ad hoc routing and NetPull (NetCenter) is that the communication endpoints in ad hoc routing are fixed, whereas NetPull tries to move the communication endpoints in order to conserve energy [10.4].