ENERGY EFFICIENT DATA DISSEMINATION IN WIRELESS SENSOR NETWORKS

A THESIS

Submitted in partial fulfilment of the
requirements for the award of the degree
of
DOCTOR OF PHILOSOPHY
in
ELECTRONICS AND COMPUTER ENGINEERING

by

TEEK PARVAL SHARMA

DEPARTMENT OF ELECTRONICS AND COMPUTER ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY ROORKEE
ROORKEE - 247 667 (INDIA)
JULY, 2009

INDIAN INSTITUTE OF TECHNOLOGY ROORKEE ROORKEE

CANDIDATE'S DECLARATION

I hereby certify that the work which is being presented in the thesis entitled **"ENERGY EFFICIENT DATA DISSEMINATION IN WIRELESS SENSOR NETWORKS"** in partial fulfilment of the requirements for the award of the Degree of Doctor of Philosophy and submitted in the Department of Electronics and Computer Engineering of Indian Institute of Technology Roorkee, Roorkee is an authentic record of my own work carried out during a period from July 2006 to July 2009 under the supervision of **Dr. R.C. Joshi**, Professor and **Dr. Manoj Misra**, Professor, Department of Electronics and Computer Engineering of Indian Institute of Technology Roorkee, Roorkee.

The matter presented in this thesis has not been submitted by me for the award of any other degree of this or any other Institute/University.

(TEEK PARVAL SHARMA)

This is to certify that the above statement made by the candidate is correct to the best of our knowledge.

Date: 20.7.09

(Manoj Misra)
Supervisor

(R.C. Joshi)
Supervisor

The Ph.D. Viva-Voce Examination of **TEEK PARVAL SHARMA**, Research Scholar, has been held on ......16.4.10......

Signature of Supervisors

Signature of External Examiner

Abstract

Wireless sensor networks (WSNs) have emerged as the recent wave of wireless technology which combines sensing, processing and communication through tiny low power and low cost Sensor Nodes (SNs). WSNs enable greatly distributed sensing across vast area through collaborating SNs. Due to very small size and disposable nature, SNs have various resource constraints. But, among all constraints, limited energy source onboard on a SN (in the form of fixed batteries) is the most challenging constraint that has created a major bottleneck in the design and deployment of WSNs. Due to this limited energy constraint, many of solutions prevalent in other ad hoc networks can not be used as such in WSNs. Since, in most of the scenarios it is not possible or economical to reclaim SNs for replenishing their energy sources, hence the only way to extend the operational lifetime of a WSN is to reduce the energy consumption while it is operational. Hence, the work presented in this thesis revolves around the central theme of reducing energy consumption within a WSN while it disseminates without compromising on granularity of required data.

For the last few years, various attempts have been made to reduce energy consumption at different layers of WSN design by targeting different set of network activities. But, among all WSN activities, communication (i.e. inter-node transmissions) is the most power hungry activity and consumes major share of overall network energy. It is known that 3000 instructions can be executed for the same cost as the transmission of one bit over 100 meters. Moreover, the numbers of inter-node transmissions are proportional to the bulk of data to be disseminated from source to sink i.e. data collecting node. Hence, in current research we target at reducing energy consumption in a WSN by minimizing numbers of inter-node transmissions caused by voluminous data. Reduction in inter-node transmissions is achieved by various ways, like treating data in-network to eliminate undesired data present in the sensed data streams generated at SNs; by avoiding generation of unnecessary (i.e. redundant) data at SNs by estimating the current behaviour of an event and setting sensing frequency dynamically; by developing an effective clustering and path setup scheme that yields smaller paths between source-sink pairs than set by existing methods; and by developing a caching strategy which increases proximity of the data nearer to sink which reduces query traversal path.

First part of the thesis presents a novel Two-Ways Sliding Window (TWSW) data filtering scheme that exploits correlation among different data items at source SNs and at

cluster head nodes (CHNs) to block/suppress undesired data transmissions towards sink. Redundant Data Filtering (RDF) and Spurious Data Filtering (SDF) are two basic components of TWSW data filtering that block data which either vary insignificantly from previously disseminated data or vary beyond maximum legitimate limit. Redundancy in data may be caused by different closely spaced SNs having same views of an event during a given sensing interval or by a SN observing same state during consecutive sensing intervals. Spurious data may be caused by temporary abnormal conditions (not related to event) prevailing around few SNs or due to malfunctioning of node software/hardware or caused by an adversary etc. TWSW efficiently isolates such unwanted data items and drops them without forwarding any further. Analytical study and simulation experiments have shown that little in-network processing done on data in the form of TWSW data filtering results in significant reduction in energy consumption by avoiding costlier and unnecessary inter-node transmissions.

Second part of the thesis presents an approach called as Adaptive Sensing and Dynamic Data Reporting (ASDDR) which helps SNs to modify their sensing frequencies dynamically to adapt to changing behaviour of an event. Since, in most cases the rate of change in event behaviour is not consistent and is highly unpredictable, hence adjudging optimal sensing frequency at the start of network operation is extremely difficult. Improper sensing frequency results in either a lot of unnecessary sensing in relatively static event scenarios or may miss many important points of observation in more dynamic scenarios. Therefor, there is need for schemes like ASDDR that sets sensing frequency according to current event behaviour.

Like TWSW data filtering, ASDDR exploits processing capabilities of nodes to treat data in-network. ASDDR basically augments TWSW data filtering with additional functionality in the form of a novel algorithm that runs at CHNs. It analyzes the flow of data reports received from SNs (i.e. reports that crossed TWSW data filtering check) to infer the current rate of change in event behaviour. Accordingly, it computes corrective adjustment values for active SNs to modify their sensing frequencies. Thus, sensing frequencies at SNs adapts automatically to event behaviour without manual intervention. ASDDR approach is very simple and easy to implement although very effective as revealed by simulation study.

Third part of the thesis explains an efficient virtual grid formation/clustering strategy known as Grid Based Data Dissemination (GBDD). GBDD results in smaller paths between source-sink pairs and handles sink/event movements effectively. Apart from reducing inter-node transmissions, the other motive behind developing GBDD is to devise a grid formation and clustering scheme that is tailor made for further proposed Cooperative Caching scheme for WSNs. GBDD exploits location awareness and dual-radio modes of SNs to construct a virtual grid of square sized cell over entire sensor field. The size of grid cell is determined by dual radio range (i.e. high power and low power) of radio transceivers on SNs. The intuition behind this is that nodes use high power radio to transmit and receive data and queries, whereas other network management activities can be handled by low power radio.

GBDD ensures automatic start and termination of the grid construction process as well as path setup between a source and a sink. Virtual grid created helps in defining clusters, setting better paths between source-sink pairs and handling movement as well as multiplicity of sinks/events effectively. Simulation results reveal that GBDD gives significant improvements in terms of overall network energy savings when compared with other existing schemes.

Fourth part of thesis gives detailed Cooperative Caching (CC) scheme that solves the problem of temporarily holding large data in the network where individual SN storage is very limited. CC exploits cooperation among various SNs in a defined region to form cooperative zones in the network to form larger virtual cache. Apart from its own limited local storage, a selected coordinator node uses storages of nodes from certain region around it to realize larger cache known as cumulative cache (CMC). Based on availability of free storage locations, CC tries to cache most of data nearer to sink Thus, apart from handling excess data flow it reduces query traversal paths and hence reduces inter-node transmissions.

As part of a complete CC scheme, various modules are devised. Token Based Cache Admission Control Scheme is devised to increase data proximity nearer to sink. A Cache Discovery Mechanism to fetch copy of data item in minimum time from a location nearest to query sender is developed. To avoid replication of data cached, a Single Copy Cache Rule is also developed which ensures that only singly copy of a data item exists in the network. Finally, a least utility based Item Replacement Policy is devised so that a data item with least utility is evicted out in case of required item replacement.

Lastly, the contribution made in the dissertation is summarized and scope for future work is outlined.

Acknowledgements

I would like to express my deepest gratitude to my learned supervisors Dr. R.C. Joshi and Dr. Manoj Misra for their encouragement, painstaking supervision, innovative suggestions and invaluable help during the entire period of my Ph.D. Their insights and capability to judge things beyond text have strengthened this study significantly. They have been instrumental in shaping my approach towards accomplishing a task in hand and I learned a great deal from them not only in research but also about aspects touching other aspects of life. I feel privileged to have worked under their supervision.

The cooperation and help extended by the Head and faculty members, Department of Electronics and Computer Engineering, Indian Institute of Technology Roorkee is gratefully acknowledged. I also want to thank my research committee members for providing insightful and constructive comments. The help rendered by technical staff of the department is heartily acknowledged.

I am highly obliged to the Director, National Institute of Technology Hamirpur (H.P.) for sponsoring me for Ph.D program. I am also grateful to my colleagues at National Institute of Technology Hamirpur (H.P.) for their direct or indirect support. Special thanks to AICTE for providing opportunity to be a part of IIT Roorkee by introducing QIP program. Thanks to QIP coordinator and staff at IIT Roorke for being extremely supportive in resolving any related issue.

I wish to convey my deep appreciation to my fellow research scholars whose company itself has been a great pleasure and a real help. Discussion and debate on a cup of tea really helped in thinking innovatively. I thank all friends at IIT Roorke who made me feel at home while away from family for three years. The moments spent with friends playing volleyball and badminton are unforgettable. Thanks friends for sharing joyful moments.

I have no words to express appreciation for my extended family for their understanding and support during these years. I could not reach the important milestone of my life without their support and encouragement.

I owe a debut gratitude to my parents who brought me up to be a confident individual and showered unconditional love on me. The values you instilled in me give me strength and make me a better person.

Most importantly, my sincere appreciation and gratitude to my beloved wife Vibha, who held the key to my success. Her patience and encouragement kept my motivation up. Despite all odds and when needed most, she intelligently managed all family matters as well as two small kids along with her job. I know three years is not a small period to be alone and you never showed. My sincere apology and love to my sweethearts Dev Kriti (daughter) and Shivansh (son) whom I kept waiting for years. Believe, it brought tears to my eyes many times and I missed a lot.

And above all, I am thankful to the Almighty whose divine grace gave me the required courage, strength and perseverance to overcome various obstacles that stood in my way.

Teek Parval Sharma

Contents

Chapter 3 Two-Way Sliding Window (TWSW) Data Filtering 51

Chapter 4 Adaptive Sensing and Dynamic Data Reporting (ASDDR) 85

Chapter 5 Dual Radio based Grid Construction and Data Dissemination 103

Chapter 6 Cooperative Caching **131**

List of Abbreviations

ACM	Active Cache Manager
ADZ	Alternate Dissemination Zone
APN	Access Point Node
ASDDR	Adaptive Sensing and Dynamic Data Reporting
CAZ	Cache Zone
CC	Cooperative Caching
CH_{buffer}	Cluster Head Buffer
CHDI	Cluster Head Dissemination Interval
CHN	Cluster Head Node
CIT	Cache Intention Token
CMC	Cumulative Cache
CN	Common Node
CN_{buffer}	Common Node Buffer
CNDI	Common Node Dissemination Interval
CNSI	Common Node Sensing Interval
CoCa	Cooperative Caching
CP	Crossing Point
CT	Cache Token
DD	Directed Diffusion
DN	Dissemination Node
EC_th	Energy Consumed computed Theoretically
EC_wf	Energy Consumed With Filtering
EC_wof	Energy Consumed Without Filtering
FDN	First Dissemination Node

FDSA	Fresh Data Sensing and Aggregation process
GBDD	Grid Based Data Dissemination
IDN	Immediate Dissemination Node
J	Joules
LEACH	Low Energy Adaptive Clustering Hierarchy
NoCoCa	No Cooperative Caching
RDF	Redundant Data Filtering
R_H	Radio transmission range in high power mode
R_L	Radio transmission range in low power mode
SDF	Spurious Data Filtering
SDN	Source Dissemination Node
SN	Sensor Node
TTDD	Two-Tier Data Dissemination
TTL	Time-to-Live
TWSW	Two-Way Sliding Window
W	Watts
WSN	Wireless Sensor Network

List of Figures

List of Tables

Chapter 1

Introduction and Problem Statement

1.1 Overview

Recent advances in micro-electro-mechanical systems (MEMS) and wireless communication have led to the development of low-cost, low power, multifunctional sensor nodes (SNs) that are small in size and communicate untethered in short distances [63] [62] [26]. These nodes can sense various environmental conditions such as temperature, sound, vibration, pressure, motion or pollutants etc. SNs when deployed randomly over vast area, collaborate among themselves through limited wireless communication capability to form an ad hoc network known as Wireless Sensor Network (WSN). A WSN is a self-organizing network that does not need user intervention for configuration or setting up routing paths. Therefore, WSNs can be used in virtually any environment, even in inhospitable terrain or where the physical placement is difficult. WSNs can combine various readings and computations over a very large area of observation to impart aggregated values in different formats and with different observed parameters. Thus, they make it possible to monitor real world events to an unprecedented level of granularity.

Wireless SN is a computing and sensing device with very stringent hardware and software constraints. Bulk deployment of SNs puts straightaway constraint on its size and cost. As shown in Figure 1.1, a SN is made up of four basic components: a sensing unit, a processing unit, a transceiver unit, and a power unit. It can comprise optional application-dependent components such as a location finding system, and mobilizer. Sensing units are usually composed of two subunits: sensors and analog-to-digital converters (ADCs). The analog signals produced by the sensors based on the observed phenomenon are converted to digital signals by the ADC, and then fed into the processing unit. The processing unit,

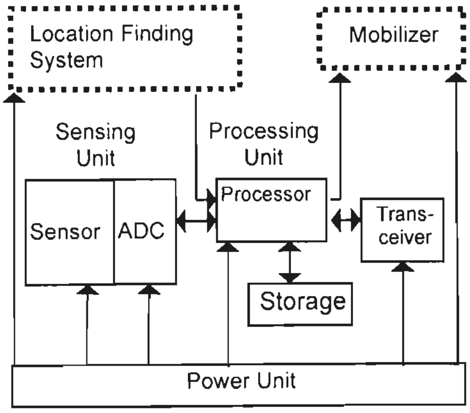


Figure 1.1. Basic Components of a Sensor Node

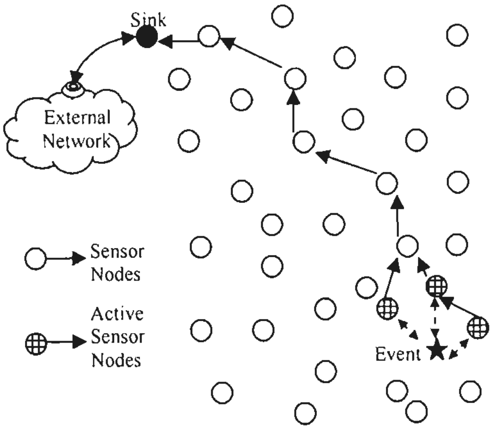


Figure 1.2. WSN Deployment Scenario

which is generally associated with a small storage unit, manages the procedures that make the SN collaborate with the other nodes to carry out the assigned sensing tasks. A transceiver unit connects the node to the network. One of the most important components of a SN is the power unit. There are also other subunits that are application-dependent. Most of the sensor network routing techniques and sensing tasks require knowledge of location with high accuracy. Thus, it is common that a SN has a location finding system. A mobilizer may sometimes be needed to move SNs when it is required to carry out the assigned tasks. All of these subunits may need to fit into a matchbox-sized module or even smaller. Now a days heterogeneous spectrum of SNs has emerged, ranging from small devices with limited hardware resources, such as the Ambient SmartTag [63], to powerful nodes approaching the capabilities of an embedded computer, such as IMote2 [117].

Randomly deployed SNs organize themselves into a network, sense real world phenomena and forward observed measurements back to base stations or user gateways. As shown in Figure 1.2, sensing and data delivery towards a sink may include a large number of wireless hops among the networked set of small, resource-limited SNs. This characteristic makes the problem of data dissemination in WSNs (WSN) become non-trivial. As a result of their operational environment, energy and memory constraints, SNs are subject to frequent failures. Making sensor networks more reliable and robust in practice is very important to enable large scale deployments and accurate measurements. Nodes need to heal the network autonomously to ensure that critical information is delivered promptly despite network changes.

1.2 Motivation

The real motivation for present study comes directly from the unique challenges in the design of WSNs and their application domain. To identify unique nature of a WSN, next we compare WSNs with mobile ad hoc networks (MANETs), give their application domain and elaborate its unique research challenges.

1.2.1 Comparison Between WSNs and Mobile Ad Hoc Networks (MANETS)

WSNs are similar to mobile ad-hoc networks (MANETs) in that both involve multi-hop wireless communications. Nodes in both networks can act both as data source as well as router as there is no central controlling station. In both networks nodes' energy consumption must be minimized, however WSN imposes far stringent energy constraint due to deployment in harsh or inaccessible terrains. However, the nature of the applications and routing requirements for the two are significantly different in several respects. They differ in many aspects summarized as:

- The hardware and software resources of nodes in WSNs are far too limited/stringent than nodes in MANETs. This attributes to the size of a node in WSN which is intended to be as small as possible.
- The numbers of nodes and their deployment density in WSNs are much higher than in MANETs. Due to sheer large number and disposable nature of nodes, the cost of the node must be very low.
- Mode of communication in a sensor network is from multiple data sources to a data recipient/sink i.e. a reverse-multicast, while in MANETSs communication is between any pair of nodes. Data flow from multiple sources to sink can be triggered either by occurrence of an event or in response to a query(s) from sink(s) or periodically after some fixed interval as programmed in SN.
- Unlike MANETs, in WSNs data generated at multiple sensors is about common phenomena, hence there is likely to be some redundancy in the data being communicated by the various sources in sensor networks.
- Nodes in WSNs are more prone to failure due to hardware/software failure or due to certain environmental conditions.
- One of the major resource constraints in WSNs is limited energy source. This constraint is also in MANETs, but situation is much worse in WSNs, where nodes in most deployments can not be recharged or replaced. Since WSNs are deployed

for very long observation periods, energy resources have to be managed even more carefully.

- Due to sheer bulk of nodes and overhead, nodes in WSN do not have any identification number (such as IP address), where as nodes in MANETs have. This makes routing in WSNs a real challenge.

1.2.2 Applications of Wireless Sensor Networks

Wireless technology has changed the way we interface with physical world. For information retrieval, an unimaginable ease of access has been achieved. For example, while on the move, a wireless device user can retrieve services at new location based on certain discovery mechanisms [43]. Although, sensors have long been used for military, research, process control and weather forecasting, but WSNs have changed the way we interface with the physical world. Instead of very few conventional sensors being used, WSN densely deploys many small nodes. These tiny distributed nodes sense the phenomenon to reveal more granular readings which can suitably be aggregated and interpreted. Different domains where WSNs find real applications are:

Habitat monitoring [3]: Monitoring habitat has proved very difficult task due to various factors like it require continuous monitoring for very long periods due to slow change in observed parameter, involve hostile geographical or geophysical locations, require very large area of interest to be observed simultaneously, generates huge volume of data which need to be analyzed for conclusions and human fatigue. However, using WSNs large numbers of sensors are randomly deployed over an area to gather sensor readings and to transmit them unattended to a central point for processing. WSNs have made it possible to take localized readings to the finest scales and resolutions by deploying enormous number of SNs networked together, which otherwise would have not been possible by traditional instruments. Deployed WSN can be active for long period of observation continuously monitoring the habitat without. Movements of birds, animals or insects can be traced and their activities can be observed to understand their behavior.

Military: Ad hoc deployment, self-organization and fault tolerance makes sensor networks very useful for military applications. Since these low-cost, disposable nodes are densely deployed, malfunctioning or destruction of few of these nodes does not hamper the military operation, which in case of traditional sensors would have been disastrous. Some of military applications of sensor networks are monitoring friendly forces, equipment and ammunition; battlefield surveillance; reconnaissance of opposing forces and terrain; targeting; battle damage assessment; and nuclear, biological and chemical (NBC) attack detection and reconnaissance.

Environment Monitoring: Sensor networks find its natural applications in many environmental areas because a large area of interest over a long period of time is observed, for example, precision agriculture. Sensors deployed in tens of thousands or more can take observations of very small zones and aggregated values through appropriate mathematical model can give clear readings of the entire observation field to the finest granularity and precision. Recent environmental applications include weather forecasting, forests fires detection, flood detection, Tsunamis detection, pollution study, geological studies and planetary exploration.

Industrial Applications: SNs can collaborate to determine and prevent potential hazardous/disastrous conditions by alerting or taking appropriate action. For example, in the oil and gas industry, dangerous situations may arise by storing incompatible substances in close proximity of each other or reaching maximum storage volume threshold for hazardous substances [88]. Also, temperature measurements over long spans at different parts of the industry or machining area can be benefited by carefully deploying wireless SNs.

Home Applications: Sensors can be built into almost all home appliances and furniture. They can be built in to refrigerator, television, washing machine, microwave oven, main power switch, at the roof, on the ceilings etc. WSN so formed can interface with the room server, which ultimately can be connected to the local network or internet. These WSNs can then be used to monitor and control various activities like automatic switching of microwave oven, refrigerator, air conditioner and television etc.; fire alarm; theft detection, remotely controlling home appliances and switches, kids activities monitoring etc.

Health Monitoring: WSNs are perceived to play major role in health care service. Sensors can be used in hospitals to monitor various activities of the patients, doctors and other staff members to optimize the time and service. Sensors can be attached to patients and their attending staff of the hospital to know the status of each other. With the size of the sensors reducing enormously with time, sensors can be used with in the body of the patient to monitor and control various symptoms and diseases.

Traffic Control: Sensors used on road crossings and other locations along the road forms a wireless network and localized readings when aggregated together give a complete view of the traffic with in a city. This view helps in regulating the traffic by suitably fixing the timings of the controlling lights at crossings. More importantly, the timings can be adjusted dynamically depending upon the traffic load in each segment so that time can be optimized. Accidents, over speeding, crossing to wrong lanes etc. can also be monitored at centralized location.

Finally, sensor grid has been a newest revelation in the WSN area that extends the grid computing paradigm (compute grid and data grid) to the sharing of sensor resources in WSNs to provide seamless access to a wide variety of resources in a pervasive manner [55]. A sensor grid is the result of the integration of WSNs with the conventional wired grid fabric.

1.2.3 Research Challenges in Wireless Sensor Networks

Resource constraints (hardware and software), ad hoc deployments, huge amount of data generated, limited communication channel and sheer numbers of nodes pose unique challenges in the design of WSNs. WSN design challenges can broadly be categorized as:

Limited Energy: SNs have scarce energy supply in the form of battery and hence energy consumption is a primary metric to be considered while designing a WSN data dissemination strategy. Often the battery of a SN is not rechargeable and the need to prolong the lifetime of a SN has a deep impact on the networking architecture and data dissemination approach. Among all networking operations communication is the major source of energy consumption. Pottie and Kaiser [45] show that 3000 instructions can be executed for the same cost as the transmission of one bit over 100m.

Scalability: The density of SNs in WSN deployment can range from few SNs to few hundred SNs or even to thousands of nodes in a region. Additional nodes may be added to existing deployment at any time or similar nodes (with same hardware and software resources) may be used at two different locations with different number of nodes. Therefore, solution proposed in the form of protocol on any networking layer or interface developed must scale up with the number of nodes added to the current deployment without affecting the required Quality-of-Service.

Self Organization: In WSN topology control, basically three entities are involved: SNs, sink and event. In most early scenarios, SNs were considered static and event and/or sink could be mobile. But in recent times various applications even have SNs mobile (like nodes attached to animals or vehicles). Therefore, big challenge for WSN designers is to provide efficient self-organization capabilities to WSNs to handle mobility; adapt to change; and even exploit dynamics to increase the functionality of the network. Self organization may even be required in case nodes fail due to malfunctioning or security attack.

Heterogeneity and Interoperability: Wide varieties of SNs are becoming available in the market and Table 1.1 summarizes few of them. It is observed that nodes range from having small capabilities in terms of hardware resources such as Ambient SmartTag to nodes approaching the capabilities of embedded computer like IMote2. The functionality of WSN can be greatly enhanced in terms of service required and extending application domain if different kinds of nodes are made part of a careful deployment. Also, most of nodes in their initial design had proprietary protocols operating at 815/915GHz band. This led to interoperability issue and unless standard radios are not used, integration is extremely difficult. However recently, IEEE 802.15.4 [154] and

Table 1.1. Few commercially available sensor nodes

Platform	**Processor**	**Radio**	**RAM**	**Flash Memory**
Crossbow MICA2[156]	8 MHz Atmel ATmega128L	868/915 MHz	4kB	128kB
Crossbow IMote2 [156]	13-416 MHz PXA271 XScale	2.4 GHz IEEE 802.15.4	32MB	32MB
Crossbow TelosB [156]	8 MHz TI MSP430	2.4 GHz IEEE 802.15.4	10kB	48kB
Ambient μNode [150]	8 MHz TI MSP430	868/915 MHz	10kB	48kB
Ambient SmartTag [150]	16 MHz Intel 8051	868/915 MHz	128 bytes	4kB
Sun SPOT [155]	180MHz ARM920T	2.4 GHz IEEE 802.15.4	512kB	4MB
SensiNode Micro2420 [152]	8 MHz TI MSP430	2.4 GHz IEEE 802.15.4	10kB	256kB
XYZ [151]	1-60 MHz OKI Semiconductor ML67 ARM	2.4 GHz IEEE 802.15.4	32kB	256kB
Sentilla Tmote Mini [153]	8MHz TI MSP4	2.4 GHz IEEE 802.15.4	10kB	48kB
Sentilla Tmote Sky [153]	8MHz TI MSP4 (Additional light sensor)	2.4 GHz IEEE 802.15.4	10kB	48kB

ZigBee [157] have emerged as prominent radio standards and there is a shift towards their compliance.

Fault Tolerance: Unlike, ad hoc networks where nodes interact with humans, WSN nodes interact with harsh and hostile environment. This puts huge responsibility on the WSN designers to provide tolerance against possible node failures. Further, in most scenarios nodes once deployed are left to their own for any re-configuration, readjustments in routing path and re-clustering. Estimating initially required nodes density, decision to redeploy more nodes and designing fault tolerant protocols at different network layers is the ultimate challenge for WSN designers.

Environmental interaction: Since WSNs have to interact with the environment, their traffic characteristics can be expected to be very different from others, human-driven forms of networks. Hence, WSNs are likely to exhibit very low data rates over a large time scale, but can have very bursty traffic when something happens. Also, due to the large number of possible combinations of sensing, computing and communication technology, many different application scenarios become possible. It is unlikely that there will be generic solutions for all these potentially very different possibilities. This unpredictability in traffic patterns and application scenarios make design of protocols for MAC and Network layer a difficult job.

Handling Highly Voluminous Data: WSNs are highly data intensive networks where hundreds or thousands of nodes generate streams of data continuously and inject then into the network. Since nodes in WSN have very limited processing and storage capabilities, to handle this huge data is real challenge. The problem here is two-fold: first is to avoid loss of useful information on account of overflow due to limited memory at sink(s), source nodes or intermediate nodes and the second is to conserve network energy by eliminating useless or spurious data. Hence, there is a need to have energy efficient data dissemination approaches which regulate the flow of data from sources to sinks.

1.3 Challenges Specific to Data Dissemination in WSNs

The main aim of our work is to handle voluminous data in energy efficient way and to increase data availability, therefore next we elaborate on typical challenges involved in designing a data dissemination approach for a highly data-intensive WSN:

Data Redundancy: Due to very dense deployment and sensing of a common phenomenon (though from different locations) a lot of redundancy is observed in data generated by SNs. Redundant data may be generated by geographically close SNs during same sensing interval or by a same SN during successive sensing intervals if monitored event becomes relatively static. WSN may loose considerable amount of energy if it disseminates entire data from source SNs to sink without any check. Since communication is a most power hungry operation in WSN, to identify this redundant data and block it from further transmissions is a serious challenge.

Excess Data Flow: In most WSN deployments, sink is one of the field SNs with an additional interface to outside network. It has same limited storage constraint as other SNs have. If quite large numbers of SNs start sending data towards sink at the same time, sink may be swamped by data flow and may drop useful information. Further, sink may not immediately require entire data, rather to infer certain observations about an event it may need to interpret data from within a time window. Therefore, there is a need to store readings during this time some where in the network. One option is to utilize memories of nodes on data/query path, but these nodes are also ordinary SNs and have limited memories which get easily exhausted. To utilize memories of nodes other than those on data/query path, a separate mechanism to find proper set of nodes to be used as temporary storage and a path setup procedure are required. Without global knowledge, handling such a situation is serious challenge.

Correlation among Data: Since communication consumes very high node energy compared to processing at that node, an obvious choice is to reduce communication cost by reducing the amount of data to be disseminated to sink by performing some in-network processing on data. In-network processing may use lossy data aggregation or lossless aggregation. Design of an aggregation function is very complex task as it needs to exploit tempo-spatial correlation among data items from different nodes and at different times. Further, complexity significantly varies from scenario to scenario and moreover on the level where it is applied. For example, if aggregation function is applied at source SNs, then correlation among different sensor readings at successive intervals may need to be found. If aggregation is done at one level up i.e. at immediate cluster heads, aggregation function may be required to find correlation among different data items from different nodes and at different times.

Lack of node id: Due to sheer bulk of nodes, disposable nature and overhead involved, wireless SNs do not posses identification numbers. In absence of ids, it is not possible to use conventional address-centric routing to set or find paths for query or data based on some routing tables. For example, queries of the nature "Fetch data from node #1044" or "Forward data to node #345" are not possible. This poses a huge challenge in designing some alternate addressing schemes for WSNs. Since path set up and data/query forwarding all should work without human intervention, this further demands some alternate ways to locate the data source and sink; find optimal path between a pair of source and sink and change path completely or partially on some node failures. Various data-centric approaches have been proposed and used in recent past for WSNs where either description of the data required is flooded or with some priori knowledge geographically scoped queries are used. Still, a great deal of research is required to find better addressing schemes which adapts to diverse class of WSN applications.

Mobility: In early perception of WSNs, once SNs are deployed, they remain static for entire observation period. In most of scenarios sink was also considered static and event could move. In such cases, data/query path once set between data source (active SNs) and sink is only partially modified to adapt to moving event. But with whole spectrum of application domains, it is evident that sink is also a moving entity. Situation gets further complicated when SNs are also mobile. Maintaining continuous data delivery despite moving sink, moving event and/or moving SNs is extremely difficult task.

Sensing Frequency: In continuous monitoring scenario once WSN is deployed, SNs are programmed for a certain sensing frequency for desired sensing parameter. Active SNs keeps on sensing event with same frequency despite random changes in the characteristics of an event i.e. whether rate of change in the observed parameter is very slow or very high. However, it is quite evident that if event becomes relatively static, lot of successive readings may generate redundant data. Dissemination of this entire data without treatment may consume network energy in transmitting data which is of no use to sink. At the other side, if event becomes very active, rate of change in observed parameter becomes too high and sensing frequency may become too low to miss some information of interest to sink. Therefore, setting sensing frequency (and accordingly data reporting rate to sink from cluster heads) dynamically to adapt to the changing nature of an event is pure challenge.

Clustering or Grid Formation: Efficient clustering and/or grid formation is the base for good data dissemination approach. In WSN, nodes discover only their neighbors and lack knowledge about global topology. Deciding a clustering scheme and/or virtual grid formation scheme, is most difficult part of the data dissemination approach. Clustering and grid formation schemes must be tolerant to node failures, since any node can fail which either might be simple node in the cluster or node with in a cell of a grid, or it may be the cluster head node (CHN) or corner node of a cell of grid. In latter case, either new cluster/grid must be constructed or if possible faulty node may be replaced by some existing eligible node.

1.4 Statement of the Problem

The main objective of the present research work can be described by the statement of the problem expressed as follows:

"To formulate data dissemination strategies for a WSN to reduce network energy consumption in disseminating voluminous data from source to sink without compromising on the required granularity of data".

To achieve this over overall objective of reducing WSN energy consumption, following smaller objectives are set:

- To investigate and propose schemes which perform in-network processing on data generated at SNs to reduce its size before injecting into the network without missing legitimate/required data.
- To explore and devise ways of avoiding unnecessary/undesired event sensing to reduce data generated at SNs.
- To devise an efficient clustering and path setup scheme that results in smaller paths between source-sink pairs and handles event/sink movements efficiently.
- To investigate and propose scheme to reduce query traversal path by devising cooperative caching that increases the proximity of data nearer to sink. This is also aimed at solving the excess data flow problem.

Communication is the most energy hungry operation among all WSN activities, whereas processing at a node consumes comparatively very less energy. Therefore, intention behind setting above objectives is to reduce inter-node transmissions in a WSN, while at the same time not to compromise on granularity of data achieved through distributed sensing.

Hence, to achieve above objectives, following design goals are set:

- Devising a data filtering scheme that does some in-network processing to exploit spatial and temporal correlation among various data items generated at some active SNs. Correlation among different data items is intended to be used to find redundant and spurious data among sensed data items and thus inter-node transmissions can be reduced by filtering/blocking such undesired data.

- Exploiting in-network processing capability of SNs further to analyze the flow of sensed data streams at some appropriate nodes in a WSN to infer the current behavious of an event. Based on inference drawn, these selected nodes can help active SNs to alter their sensing frequency accordingly. Thus, the goal is to induce adaptation in the working of active SNs so that they avoid unnecessary event sensing. This will reduce bulk of sensed data generated and hence will reduce inter-node transmission
- Devising a virtual grid formation scheme for converting entire sensor field in to a square sized grid. This virtual grid is intended for conveniently defining clusters and to set optimal paths between source-sink pairs. Smaller the paths, fewer are inter-node transmissions. To decide the cell size of a grid is very vital in setting paths and we plan to use dual radio modes of SNs to decide the same. Dual radio helps in conserving energy by using low power radio instead of high power radio wherever possible for any network activity. Apart from clustering, virtual grid is intended to energy efficiently handle event and source movements without incurring much overheads.
- Proposing a caching scheme that can temporarily hold excess data flow at some appropriate points (i.e. nodes) in network. Since individual node's storage in a WSN is very small, intention is to exploit cooperation among various nearby nodes to realize much larger cooperative (and virtual) cache. We plan to develop caching strategy in such a manner that most of cached data items remains close to sink so that queries issued by sink can be serviced as early as possible to reduce inter-node transmissions. Cooperative caching will also increase data availability by avoiding dropping of data due to swamping of sink and other nodes with excess data flow.

1.5 Thesis Organization

Rest of the thesis is organized as follows:

Chapter 2 gives fundamentals and literature review of various clustering, in-network data processing and routing schemes. We then review caching schemes proposed/used for WSNs as well as some closely related schemes in MANETs. Also, we look into the recent trend of use of multi-mode SNs, especially dual radio mode SNs which is part of assumptions made in present study. At the end, research gaps are given and brief summary concludes the chapter.

Chapter 3 gives Two-Way Sliding Window (TWSW) data filtering approach devised for suppressing undesired sensed data with the intention to reduce its size before its further transmission towards sink. Chapter includes details of filtering approach, analytical study for computing energy consumption and its performance evaluation through series of simulation experiments conducted under varying system settings.

Chapter 4 presents a novel approach called Adaptive Sensing and Dynamic Data Reporting (ASDDR) which is a novel approach devised to enable active SNs to adjust their sensing frequencies according to current event behaviour. Chapter includes results of simulation experiments that reveal significant reduction in energy savings when compared to scenarios when no data filtering is used as well as scenarios when TWSW data filtering is used.

Chapter 5 presents a scheme named Grid Based Data Dissemination (GBDD) that constructs virtual grid over entire sensor field. Chapter details various design aspects of GBDD including deciding the cell size of the grid, grid construction mechanism (i.e. starting the process, creating cells and terminating the process), clustering based on grid cells, path setup mechanism and handling movement of sink(s) and/or event(s). Performance of proposed approach is evaluated against existing approaches.

Chapter 6 introduces Cooperative Caching (CC) scheme proposed to reduce query traversal path and to handle excess data flow. As part of complete cooperative caching, it includes: Token Based Cache Admission Control, Cache Discovery system, Single Copy Cache Rule and least utility based Item Replacement Policy. Performance of cooperative caching is evaluated through series of experiments where different system settings are used for varying scenarios.

Chapter 7 concludes the thesis with summary of contributions towards energy efficient data dissemination in WSNs and also gives directions for future work.

Chapter 2

Background and Literature Survey

The term data dissemination in WSNs refers to a complete process of transferring desired data from active SNs (i.e. nodes sensing event) to data collecting node i.e. sink. Routing and data dissemination terms are sometimes used interchangeably by many authors working in the area of ad hoc WSNs. However, there is a notional difference between the two; routing refers to the process of simply transferring raw packets from source to destination without bothering about its contents and no in-network processing (except routing) is done based upon their contents. Whereas, data dissemination may involve some in-network processing of data items like data fusion, data aggregation based on correlation among data items and prioritization of data items based on their contents etc. Like most of researchers working in WSNs area, we use the term data dissemination to refer to a process encompassing possible clustering, routing, in-network processing and/or caching. However, boundaries defining them may sometimes overlap and we have tried to clarify the same wherever required.

Limited energy source has been the single major constraint affecting the design of a WSN at all its layers of abstraction. Hence, energy efficiency remains an essential performance metric at all design layers of a WSN and aim has been to enhance the operational lifetime of the network. Among many, few ways to enhance the life time of a WSN are: finding optimal routes for query/data flow; devising efficient clustering schemes; load balancing among nodes [123]; by employing in-network processing like data fusion, data compression and data filtering etc. Authors in [1] [70] [83] [146] devise new MAC for WSN or propose ways to achieve energy efficiency at existing MAC layer. Unbalanced energy depletion among SNs also reduces lifespan of a WSN and hence various methods have also been proposed to distribute load by rotating roles of cluster heads, distributing computations across network and coordinating various working modes (wake-up, sleep and sensing) of SNs etc [13] [99] [15] [109] [123]. Channel allocation and bandwidth sharing is another trivial problem in such networks [124] [57].

Fundamentals, classifications and literature review of various clustering, in-network data processing (especially data aggregation), routing and caching strategies proposed/used for WSNs are presented next. Since dual radio modes of SNs have been greatly exploited in current study, therefore at the end we review work related to multi-mode SNs, especially the use of dual mode radios.

2.1 Clustering

2.1.1 Clustering Fundamentals

Clustering has always been a basic method to organize large number of objects into suitable groups in almost all fields of science and engineering. For example, at one end Lin and Gerla [22] propose a self-organizing adaptive clustering for multihop mobile radio network while Gupta *et al.* [125] give k-means based clustering scheme for large data sets. Due to very large number of SNs and possible later addition of more nodes to existing network or random failures of nodes make scalability as major design issue for WSNs. If topology is kept flat, it is extremely hard to manage and maintain WSN where substantial traffic is caused by network dynamics, very large numbers of nodes compete for limited bandwidth and setting path between source-sink pairs is tedious. Therefore, to solve the scalability problem one can abstract network topology as hierarchies of nodes. This process is commonly referred to as clustering.

Since inception of WSNs grouping SNs into clusters has been a standard practice by the research community in order to achieve the network scalability objective [15]. Every cluster has a leader, often referred to as the cluster-head (CH). Either SNs themselves choose CHs or are pre-assigned by the network designer. A CH node may be just one of the normal SNs or a node that is richer in resources. The cluster membership may be fixed or variable. CHs may further form a second tier network or may just transmit data to a sink. Also, frequent failures of nodes including cluster heads in MANETa and WSNs require certain degree of fault tolerance through election/re-election of new cluster heads and consensus among nodes [131] [158] [39]. Cardei *et al.* [84] introduce fault-tolerant topology control for a heterogeneous WSN. They propose k-degree Anycast Topology Control (k-ATC) scheme with the objective of selecting each sensor's transmission range such that each sensor is k-vertex supernode connected and the total power consumed by sensors is minimized. Such topologies are needed for applications that support sensor data reporting even in the event of failures of up to $k - 1$ SNs.

2.1.2 Clustering Benefits

Clustering helps to handle scalability by making a large network appear smaller, and a highly dynamic topology appear less dynamic [6]. These factors further helps in following:

Medium Access Control: Separate bandwidth can be allocated to each cluster and thus inter-cluster interaction and exchange of redundant massages can be avoided [21].

Flooding: In WSNs, due to absence of pre-defined data/query paths and lack of global knowledge like sink/source position, flooding remains the only option at initial stages of WSN operation to issue queries and send data. In flat topology, flooding is very expensive and suffers from many disadvantages, whereas in case of clustering flooding is restricted only with in a cluster [61].

Data Aggregation: Cluster heads get opportunities to suitable aggregate data (lossy or lossless) before transmitting it further to higher level cluster head or to sink or to next cluster head on the data path [147]. This helps in reducing the numbers of inter-node transmissions and consequently reduces network energy consumption.

Hiding topology details: Clustering also helps in hiding details of static or dynamic topology with in a cluster until nodes do not cross-over to other clusters. Dynamics of topology are restricted with in the cluster and its impact on outside nodes is minimized.

Joint Movement: Nodes with in close proximity with similar movement patterns can be identified and made as part of same cluster. This helps to minimize the impact of movement on collaborative processing and routing.

2.1.3 Clustering Requirements

In WSNs, where topology is very random and must auto-reconfigure in case of multiple node failures, the broad clustering requirements are as follows:

Reliability: While deciding size and shape of a clustering scheme, the foremost objective is that the wireless communication between various cluster heads and among all communicating nodes from any possible source to sink is reliable. It must provide a platform to realize a stable and reliable information flow using certain transport protocols [20].

Symmetry: As far as possible, all communication links must be symmetrical i.e. If I can hear you means you can also hear me. This requirement stems from the fact that in WSNs due to changed dynamics any node can become data source, data recipient, cluster head or forwarding node and these roles can interchange.

Unique node ids: Nodes in WSNs do not posses any specific IP number style identification number due to sheer number and disposable nature, but for clustering some alternate node identification mechanism must exist to define cluster membership. For example, geographic coordinates through GPS, locally temporary assigned numbers through local interactions etc. This information is useful for grouping nodes based on their location [60] [61].

Global information: The numbers of nodes within the network or the total remaining energy represent global information, which can be useful for achieving the desired clustering structure.

Synchronization: Clustering and subsequent data interpretation at cluster heads requires some synchronization mechanism to accurately carry out certain coordinated activities with in a cluster [75] [74].

Load Balancing: Since CHs do all necessary in-network processing, data buffering and data forwarding etc, it is intuitive to balance the load among them so as to get optimal performance [93] [123]. Load balancing becomes more important if CHs are picked from the available SNs and are not separate resource richer nodes. Setting equal-sized clusters is crucial for extending the network lifetime since it prevents few CHs from prematurely exhausting their energy while others are under-utilized.

2.1.4 Parameters for Clustering Comparison

In literature various parameters such as cluster convergence rate, cluster stability, cluster overlapping (i.e. whether cluster overlap allowed or not), location awareness (whether clustering scheme requires node ids or not) and support for handling node (sink, field nodes and source nodes) mobility are used to compare clustering approaches. For example if scalability is the main concern, cluster convergence rate is selected as the evaluation metric and if WSN is deployed in highly dynamic environment then clustering that support high node mobility is most desirable.

2.2 Clustering Schemes

Various clustering schemes for WSNs exist in literature with each scheme oriented towards meeting a separate performance criteria which is specified by the application in hand. Also, various classification criteria such as purpose of clustering, number of clusters, cluster convergence rate, levels of clustering, degree of mobility and complexity have been used to categorize these approaches. Since focus of our study is towards achieving energy efficiency in scalable WSNs where event and/or sink may be mobile, we choose convergence rate and support for mobility as the main criteria for classifying different clustering approaches. Hence, four main categories emerge as:

- *Variable Convergence Time with support for Mobility*
- *Variable Convergence Time without support for Mobility*
- *Constant Convergence Time with support for Mobility*
- *Constant Convergence Time without support for Mobility*

Next we discuss clustering schemes falling under each category and in the end we summarize them in a table comparing their attributes.

2.2.1 Variable Convergence Time with Mobility

Baker and Ephremides [34] [35] propose one of the earliest clustering approach LCA (Linked Cluster Approach) where node movements are handled by cluster head nodes by forming a back bone to which cluster member nodes can connect while on the move. The Objective of the proposed distributed algorithm is to form clusters such that a CH is directly connected to all nodes in its cluster and thus maximizes network connectivity. However, algorithm assumes that nodes are synchronized and access to the medium is time slot based. A node is assigned the slot in the frame that matches its ID.

First, each node broadcasts its ID and listens to transmissions of other nodes. In the next round, a node broadcasts the set of neighbors from whom it heard in previous round and thus every node will know its 1-hop and 2-hop neighbors. A node x becomes a CH if it has the highest ID among its neighbors or does not have the highest ID in its 1-hop neighborhood, but there exists at least one neighboring node y such that x is the highest ID node in y's 1-hop neighborhood. Drawback of this approach is that it yields excessive numbers of clusters. The convergence time for this clustering scheme is O(n), where n represents number of nodes in the network.

A. Ephremides *et al.* [2] refine LCA with the idea to pick a node x at random as the first CH and assign its neighbors to such first cluster. The node y with the lowest ID in the cluster is nominated as a second CH. The neighbors of y that are not reachable to x would join the second cluster. The procedure is repeated for the third cluster and so on.

Lin and Gerla [22] propose a self-organizing adaptive clustering for multihop mobile radio network which relies on a code-division access scheme for multimedia support. Nodes are organized into non-overlapping clusters. With node movement clusters are dynamically reconfigured and independently controlled. This clustering strategy provides many advantages: it provides spatial reuse of the bandwidth due to node clustering, bandwidth can be shared or reserved in a controlled fashion in each cluster and the cluster algorithm is robust in the face of topological changes.

Nagpal and Coore [115] propose CLUBS, an algorithm that forms clusters with a maximum of two hops through local broadcast and converge in a time proportional to the local density of nodes. Each node in the network chooses a random number from a fixed integer range and counts down from that number. If the count down reaches zero without interruption from any other neighboring node, it announces itself CH and broadcasts a ''recruit'' message. When a neighboring node receives the "recruit" message that comes within two-hop diameter boundary, it stops the count down, accepts the invitation and joins the cluster. A node joins a cluster is termed as ''follower'' and is no longer allowed to compete for being a CH. Follower nodes keep listening to additional recruit messages and can be follower of more than on CH. If a node that is competing to become a CH detects a collision or receives a garbled message, it becomes a follower node and assumes that multiple CHs attempted to recruit it at the same time. It can find out its CH later. The algorithm does not terminate unless all nodes in the network join some cluster as a CH or as a follower. Cooper *et al.* [32] [33] propose a family of message propagation protocols for highly mobile ad-hoc networks. The coverage of a message (the fraction of nodes that receive it), is made arbitrarily close to 1, at a moderate cost of extra message traffic.

Random competition based clustering (RCC) [77] is developed for scenario where few nodes act as backbone nodes or cluster heads and the whole subnet is formed around them. These backbone nodes periodically broadcast their beacons and ordinary nodes listen to these beacons to select the nearest backbone node to form cluster or subnet. But these backbone nodes may fail or move out of current cluster. RCC uses redundant backbone nodes so that failed or moved backbone nodes can be replaced. If two or more backbone nodes come close, only one of them keeps the role of backbone node or cluster head and others give away. RCC employs a scheme in which a node declares itself a cluster head if it detects that it is the first node in its radio range to claim candidacy for CH. This clustering mainly thrives for cluster stability. After hearing the claim which is broadcasted by the first node, neighboring nodes join its cluster as member and give up their right to be a CH. For cluster stability CH broadcasts its claim periodically. One problem with this scheme which is obvious for any distributed setup is that there might be delay in broadcast of a claim and reception of the same by neighboring nodes and as a consequence many nodes may broadcast CH claim concurrently. RCC uses random timer at nodes to start broadcast and node id for this conflict resolution, for example, node with least id may be selected as CH.

2.2.2 Variable Convergence Time without support for Mobility

Bandyopadhyay and Coyle [118] proposed an Energy Efficient Hierarchical Clustering (EEHC) algorithm for WSNs with the objective to maximize network lifetime. They propose a multilevel clustering scheme where at each level nodes aggregate data to send only aggregated values to higher level. This clustering scheme works in two stages. Initially, each SN announces itself as a CH with probability p to the neighboring nodes within its communication range and is termed as volunteer CH. All nods that are within *k* hops range of a CH receive this announcement and node is not a CH, it becomes the member of the closest cluster. Value of *k* is computed to yield minimum energy consumption and defines maximum number of hops allowed from a sensor to its cluster head. Forced CHs are nodes that are neither CHs nor belong to a cluster. If the announcement does not reach to a node within a preset time interval t that is calculated based on the duration for a packet to reach a node that is k hops away, the node will become a forced CH assuming that it is not within k hops of all volunteer CHs. In second stage, clustering is allowed to extend to any levels. For example, at second level, CHs are treated similar to nodes at first level and thus they further form clusters by repeating the same procedure as performed at first level. New cluster head aggregates values from all first level cluster heads and either transmits to sink (i.e. base station) or to higher level cluster head in case of 3-level clustering.

An energy-efficient multi-level clustering (EEMC) algorithm [161] constructs a similar k-level hierarchy. Each node is assumed to know its location coordinates. At the beginning of the cluster formation phase, the sink node collects the location information from the nodes and sends back the total remaining energy of the network and the total distance between the nodes and the sink. The probability for the first hierarchical level of cluster heads is computed using these two values. For the next level, the cluster heads broadcast the total remaining energy and the total inter-node distance in their cluster, which are used to compute the next level probability.

Wan *et al.* [108] propose an algorithm for construction of a connected dominating set of nodes, by assuming the existence of a rooted spanning tree structure already in place. Ordered pair of the tree level and the node ID acts as weight of a node. The algorithm has two phases: the construction of a maximal independent set and the construction of a dominating tree, whose internal nodes become a connected dominating set.

2.2.3 Constant Convergence Time with Support for Mobility

M. Demirbas *et al.* propose a distributed technique known as Fast Local Clustering service (FLOC) [86] which produces clusters of approximately equal size with minimum over-lap. The assumed radio model classifies nodes based on their proximity to the CH into inner (i-band) and outer (o-band). Inner nodes of the cluster are put in i-band and outer nodes in o-band. Nodes in i-band suffer very little interference while communicating with the CH, whereas message from o-band nodes may be lost due to interference. FLOC favors i-band membership in order to increase the robustness of the intra-cluster traffic.

Complete transition diagram to show working of the algorithm is given in Figure 2.1 and is redrawn from [86]. A node waits for some random duration to receive an invitation from any potential CH. If during this period no invitation is heard, it becomes a candidate CH and broadcasts its candidacy (transition 1). A node ''k'' which is already a i-band member of a cluster Ck when hears this candidacy message, reply back candidate CH to inform about such membership. The candidate CH realizes the conflict and joins Ck as an o-band node (transition 3). Otherwise, if no such conflicting message is received, it becomes a CH and invites members to its cluster (transition 4). An idle node joins a cluster as an o-band node (transition 5) if it does not receive an invitation from a closer CH (transition 2). However, that decision can be changed, if the node later receives an invitation from a closer CH, i.e. the node switch its membership to a better cluster (transition 6). FLOC scales very well converging in a constant time is O(1), regardless the size of the network. This clustering algorithm also exhibits self-healing capabilities since o-band nodes in one cluster can switch to an i-band node in another cluster.

Algorithm for Cluster Establishment (ACE) [47] is a clustering algorithm where a node assesses its capability to become CH and defers itself if it does not have. Also, each CH periodically checks the ability of its neighbors for being a CH and decides to step

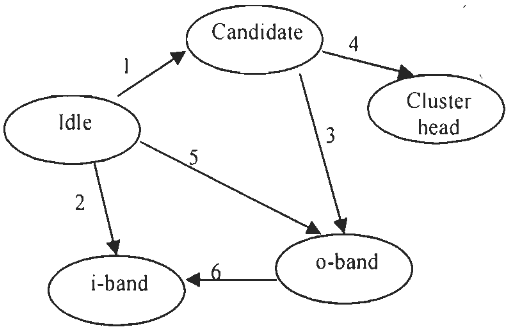


Figure 2.1. Transition diagram for FLOC

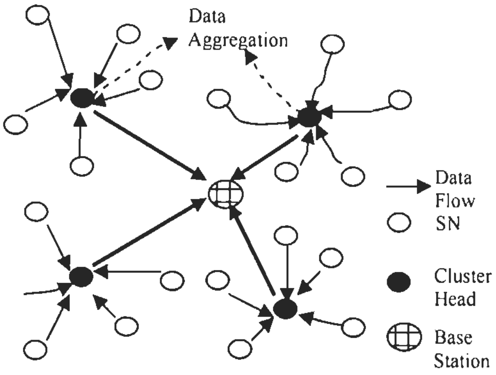


Figure 2.2. Clustering in LEACH

down if one of these neighbors has more followers than it does. A node that has the largest number of followers and the least overlap with existing clusters will be considered as the best candidate for CH. The overall effect would appear as clusters are applying a repulsive force to spread out and reduce their overlap. A node spawns of new cluster when it decides to become a CH. It broadcasts an invitation message to recruit its neighbors. Upon getting the invitation, a neighboring sensor joins the new cluster and becomes a follower of the new CH. At any moment, a node can be follower of more than one clusters. However, the node can be a loyal follower, i.e. a member, of only a single cluster.

2.2.4 Constant Convergence Time without Support for Mobility

Low Energy Adaptive Clustering Hierarchy (LEACH) [148] is one of early energy efficient and most popular clustering algorithms for WSNs and is shown in Figure 2.2. It has following main properties: self-organizing, adaptive clustering, evenly distributes load among the sensors, dynamically forms clusters, randomly rotates cluster heads after each round, cluster-heads directly communicate data with the base station and application-specific data processing, such as data aggregation. LEACH has two phases: setup phase and steady state phase.

Setup Phase: Each sensor chooses a random number m between 0 and 1. If m < T(n) for node n, the node becomes a cluster-head where

$$T(n) = \begin{cases} \dfrac{P}{1 - P[r * \mathrm{mod}(1/P)]} & if \ n \in G \\ 0 & otherwise, \end{cases}$$

where P is the desired percentage of CHs, r is the round number, G is the set of nodes that have not been CHs during the last 1 / P rounds.

. A CH advertises itself to neighbors using a CSMA MAC. Surrounding nodes decide which cluster to join based on the signal strength of these messages.

Cluster heads assign a TDMA schedule for their members. As being a CH node is much more energy intensive than being a non-CH node, this requires that each node takes its turn as CH. The role of being a CH is rotated periodically among the nodes of the cluster in order to balance the load.

Steady-state phase: All source nodes send their data to their CHs. CHs perform data aggregation/fusion through local transmission and send them back to the sink using a direct transmission. After a certain period of time, CHs are selected again through the set-up phase.

Hybrid Energy-Efficient Distributed Clustering (HEED) [100] is a clustering algorithm which works in a distributed manner where CH nodes are picked from the deployed sensors. CH selection is based upon energy and communication cost both whereas LEACH selects cell-head nodes randomly. Only sensors that have a high residual energy can become cell-head nodes. In HEED, two nodes with in each others transmission range rarely become CHs. This means CHs are properly distributed throughout the sensor field. Also, each node is mapped to exactly one cluster and can directly communicate with its CH. A key feature of this approach is that it exploits the availability of multiple transmission power levels at SNs.

Initialization phase: An initial percentage of CHs among all n nodes, C_{prob} is set, assuming that an optimal percentage cannot be computed a priori. Then for each SN, its probability CH_{prob} to become CH is computed which is used to limit the initial CH announcements to the other sensors. CH_{prob} is calculated as follows:

$$CH_{prob} = C_{prob} * E_{residual}/E_{max},$$

where $E_{residual}$ is the current energy in the sensor, and E_{max} is the maximum energy, which corresponds to a fully charged battery.

CH_{prob} is not allowed to fall below a certain threshold p_{min}, which is selected to be inversely proportional to E_{max}.

Repetition phase: During this phase, every SN finds a CH with which it can communicate with least transmission power (cost). If it does not hear from any CH, it elects itself to be a CH and sends an announcement message to its neighbors informing them about the change of status. Finally, each SN doubles its CH_{prob} value and goes to the next iteration of this phase. It stops executing this phase when its CH_{prob} reaches 1. Therefore, a node can act as tentative cluster head with CH_{prob} value less than one or permanent CH if CH_{prob} value reaches one.

Finalization phase: During this phase, each SN either picks the least cost CH or pronounces itself as CH.

Ding *et al.* [103] have proposed 'Distributed Weight-Based Energy-Efficient Hierarchical Clustering' scheme similar to HEED but with an intention to generate more balanced cluster sizes and optimize the intra-cluster topology. Each sensor calculates its weight after locating the neighboring nodes in its area. The weight is a function of the sensor's energy reserve and the proximity to the neighbors. In a neighborhood, the node with largest weight would be selected as a CH through election process and the remaining nodes become members by joining the CH hierarchy. The clustering process terminates in O(1) iterations, and does not depend on network topology or size.

Multi-hop Overlapping Clustering Algorithm (MOCA) [11] is a clustering algorithm for organizing the SNs into overlapping clusters and intended to facilitate inter-cluster routing, topology discovery and node localization and recovery from CH failure. It guarantees some degree of overlap among clusters to facilitate many such applications. The goal of the clustering process is to ensure that each node is either a CH or within k hops from at least one CH, where k is a preset cluster radius.

Mao *et al.* [92] propose a novel clustering schema EECS for WSNs, which better suits the periodical data gathering applications. The approach elects cluster heads with more residual energy through local radio communication while achieving well cluster head distribution and proper load balancing is performed among cluster heads.

2.2.5 Comparison of clustering schemes

Table 2.1 compares main clustering approaches based on the

- time they take to converge towards completing the clustering process
- clustering overlap defining whether various clusters share some SNs as their member nodes and
- symmetry in forming equal sized clusters.

Table 2.1. Classification of clustering schemes

Category	Clustering Scheme	Convergence Time	Clustering Overlap	Cluster Symmetry
Variable Convergence Time with support for Mobility	LCA Baker, Ephremides [34] [35]	O(n)	No	Medium
	Adaptive Clustering Lin , Gerla [22]	O(n)	No	Medium
	CLUBS Nagpal, Coore [115]	O(n)	Yes	Medium
	RCC Xu, Gerla [77]	O(n)	No	Good
Variable Convergence Time without support for Mobility	EEHC Bandyopadhyay, Coyle [118]	$O(k_1+k_2+\ldots+k_h)$ h=levels of hierarchy	No	Medium
	EEMC Yan *et al.* [161]	O(log log n)	No	Ok
	Wan *et al.* [108]	O(n)	Yes	Ok
Constant Convergence Time with support for Mobility	FLOC M. Demirbas *et al.* [86]	O(1)	No	Good
	ACE Chan, Perrig [47]	O(d),d=estimated average number of neighbors of a node in the network	Yes	Good
Constant Convergence Time without support for Mobility	LEACH Heinzelman *et al* [148]	O(1)	No	Ok
	HEED Younis, Fahmy [100]	O(1)	No	Good
	DWEHC Ding *et al.* [103]	O(1)	No	Very Good
	MOCA Youssef *et al.* [11]	O(1)	Yes	Good

2.3 In-network Data Processing

In WSNs, aggregation techniques and routing protocols are not independent, rather interdependent. Routing protocol design takes into consideration the targeted data aggregation at some network nodes and accordingly decides packet routing mechanism.

Similarly, while designing aggregation technique the routing protocol used underneath plays a vital role. However, for clarity of the strategy adopted, an attempt is made to give each one separately. In-network aggregation is explained in sub-sections to follow of this section and routing protocols are given in Section 2.4.

2.3.1 In-network Data Aggregation

In WSNs, data generated by different sensors can be jointly processed while being forwarded toward the sink. This can be accomplished by all or few of following actions:

a) Fusing together sensor readings related to the same event.

b) Locally processing raw data before injecting into the network.

c) Removing redundancy from the generated data before transmission.

Data aggregation is the simplest type of in-network processing which combines data from different sources or nodes in to a single entity. Data aggregation techniques are closely related to the way data is gathered at SNs as well as how packets are routed through the network. Data Aggregation has significant impact on energy consumption and overall network efficiency. However, data size reduction through in-network processing should not diminish required granularity of information about the monitored event. Also, apart from reducing many network overheads, it should be useful in enhancing network lifetime. According to [36], "*in-network aggregation is the global process of gathering and routing information through a multihop network, processing data at intermediate nodes with the objective of reducing resource consumption (in particular energy), thereby increasing network lifetime*".

2.3.2 Types of In-network Data Aggregation in WSN

Data aggregation can be classified as in-network aggregation with data size reduction and in-network aggregation without data size reduction.

In-network Aggregation with data size reduction: This is a process of combining data from different sources to a data unit which is much smaller than the total size of individual data from different sources. The aim is to reduce the size of information to be sent over the network. For example, if we take a WSN deployed for sensing temperature over a very large area, reading from SNs under same cluster can be averaged at cluster head and thus only single data packet is transmitted further in the network hierarchy towards sink. However, this type of aggregation some times looses granularity of sensed observations achieved through distributed sensing. If after data aggregation operation sink requires each individual reading for analysis and it is not possible to perfectly reconstruct all of the original data, then it is termed as lossy aggregation. The type of aggregation function applied actually makes data aggregation as lossy or lossless. All aggregation functions are not lossy.

In-network Aggregation without data size reduction: If data packets from different sources or nodes are combined in to one packet without any processing (like average, min, max, median etc) on it. For example, suppose SNs are programmed to measure two different event parameters namely temperature and pressure. A cluster head node receives a packet comprising temperature reading from one node and pressure reading from another. These two readings can not be averaged but surely can be put into one larger packet as such and instead of two separate packets cluster head transmits single packet. This results in significant reduction in network overheads and preserves the granularity of data by making each individual packet from source available to sink for further analysis. However, the bulk of individual data reading collected at sink would be huge and it becomes worse if numbers of parameters observed are more. Some applications may log entire data in some storage space for a particular period of observation and later this offline data may be interpreted by applying suitable information extraction techniques like [114].

2.3.3 Data Aggregation Strategies

If there are K sources all close to each other and far away from the sink, the combination of their data into a single packet leads, on average, to a K-fold reduction in transmissions with respect to the case where all data are sent separately. Generally, the optimal joint routing and compression structure is a Steiner tree, which is known to be NP hard. However, there exist polynomial solutions for special cases where the information sources are close to each other. The authors in [122] propose a model to describe the spatial correlation in terms of joint entropy. They analyze a symmetric line network with different degrees of correlation among neighboring nodes. For the uncorrelated case, the authors show that the best routing strategy is to forward packets along shortest paths. In contrast, in case of completely correlated information, the best strategy is to aggregate data as soon as possible. After that, a single packet (formed by the aggregated data) is sent to the sink along the shortest path. Zhu *et al.* [166] study the impact of data correlation on the energy expenditure of data distribution protocols. They focus on various energy-aware data aggregation trees under different network conditions, such as node density, source density, source distribution, and data aggregation degree.

A further tree-based aggregation algorithm that exploits data correlation is presented in [107]. It is based on shallow Length Tree (SLT) that unifies the properties of Minimum Steiner Tree (MST) and Shortest Path Tree (SPT). In an SLT, the total cost of the tree is only a constant factor larger than that of the MST, while the distances (and delays thereof) between any node and the sink are only a constant factor larger than the shortest paths. Cristescu *et al.* [110] analyze aggregation properties of a tree structure that is based on an SPT of nodes close to the sink node, while nodes that are further away are connected to the leaves of the SPT via paths found by an approximation algorithm for the traveling salesman problem. In [87], [71], [46] authors study the ways where sink organizes routing paths to evenly and optimally distribute the energy consumption while favoring the aggregation of data at the intermediate nodes. Dasgupta *et al.* [71] use linear programming to compute aggregation topologies by taking into account the residual energy of each node.

Al-Karaki *et al.* [69] investigate which nodes in the network can be exploited as aggregation points for optimal performance. They present exact and approximate algorithms to find the minimum number of aggregation points in order to maximize the network lifetime. Algorithms use a fixed virtual wireless backbone that is built on top of the physical topology. Further, they study tradeoffs between energy savings and the potential delay involved in the data aggregation process. In [59] and [44] the focus is on the nodes that should be entrusted with the transmission of the sensed values, whereas in [140] the emphasis is put on the proper scheduling of sleeping/active periods. Optimal paths are calculated in a centralized manner at the sink by exploiting different assumptions on the data correlation and selecting the best aggregation points by means of cost functions [56].

More recently, tree-based schemes for real time or time-constrained applications have been proposed [50], [160], and [64]. Further, Gupta *et al.* [51] propose an approach that relies on the construction of connected dominating sets. These sets consist of a small subset of nodes which form a connected backbone and whose positions are such that they can collect data from any point in the network. Nodes that do not belong to these sets are allowed to sleep when they do not have data to send. Some rotation of the nodes in the dominating set is recommended for energy balancing. Further algorithms that exploit/study in-network aggregation can be found in [59], [140], [69], [44].

2.4 Data Routing in WSNs

Limited energy source, small bandwidth and typical deployment of large number of SNs pose many challenges to the design and management of WSNs. These constraints necessitate energy savings at all layers of networking protocol stack. Among many design challenges, routing in WSNs is very vital due to several unique characteristics of WSNs that distinguish them from contemporary wireless ad hoc networks. Such few unique characteristics of WSNs that affect routing are:

- SNs are highly constrained in terms of transmission power, energy, processing capacity and storage.
- Due to the absence of node identification number, routing based on classical IP-based global addressing scheme cannot be applied to WSNs. This necessitates other style of addressing schemes which hugely affects the design of routing scheme.

- Unlike other networks, almost all applications of WSNs require flow of sensed data from multiple sources to a particular destination i.e. sink.
- Nodes in close vicinity may generate data with significant redundancy.
- Nodes are prone to frequent failures and must result in network partitions or inaccessible zones. Routing technique used should prolong network lifetime by embedding fault tolerance and energy dissipation balance among nodes.

Although, many parameters exist to categorize routing protocols for WSNs, but here use data aggregation to classify whole spectrum of routing techniques. Accordingly, two broad categories emerge as *routing which employs certain in-network data aggregation and routing which does not employ any in-network data aggregation.*

2.4.1 Routing with In-network Data Aggregation

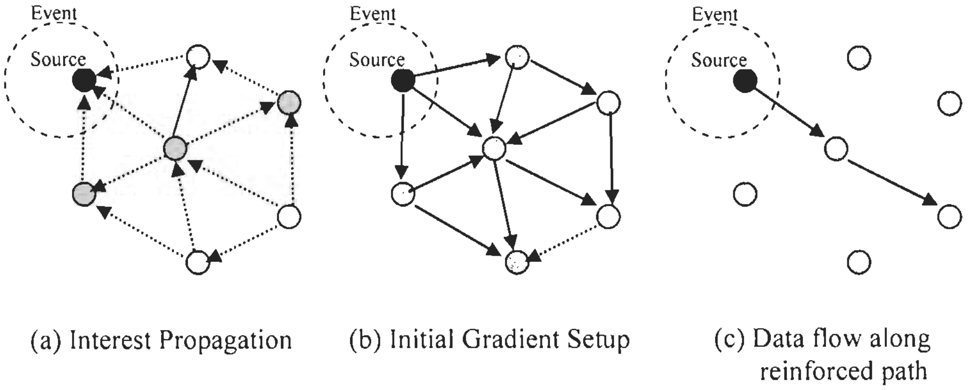
Directed Diffusion (DD) [17]: Intanagonwiwat *et al.* proposed directed diffusion as one of the most popular and scalable approach for energy efficient data routing and in-network processing for WSNs. It is a data-centric and application-aware paradigm in which data generated by SNs is named by attribute-value pairs. Also, queries do not follow address-centric approach, such as "*Fetch data from node #678*", rather use different data naming system to flood description of the task or required data. Given a set of tasks supported by a sensor network, selecting a data naming scheme is the first step in designing directed diffusion for the network. For example, a vehicle tracking task might be described as:

type = four wheeled vehicle // *detect vehicle location*
interval = 20 ms // *send events every 20 ms*
duration = 10 seconds // *for the next 10 seconds*
rect = [-100, 100, 200, 400] // *from sensors within rectangular region*

The task description specifies an interest for data matching the attributes. For this reason, such a task description is called an interest. The data sent in response to interests is also named using a similar naming scheme. Thus, for example, a sensor that detects a wheeled vehicle might generate the following data:

type = wheeled vehicle // *type of vehicle seen*
instance = truck // *instance of this type*
location = [125, 220] // *node location*
intensity = 0.6 // *signal amplitude measure*
confidence = 0.85 // *confidence in the match*
timestamp = 01:20:40 // *event generation time*

Like any data centric paradigm, DD combines data coming from different sources en-route by eliminating redundancy, minimizing the number of transmissions, thus saves network energy and prolongs its lifetime,. Unlike traditional end-to-end routing, it finds



(a) Interest Propagation (b) Initial Gradient Setup (c) Data flow along reinforced path

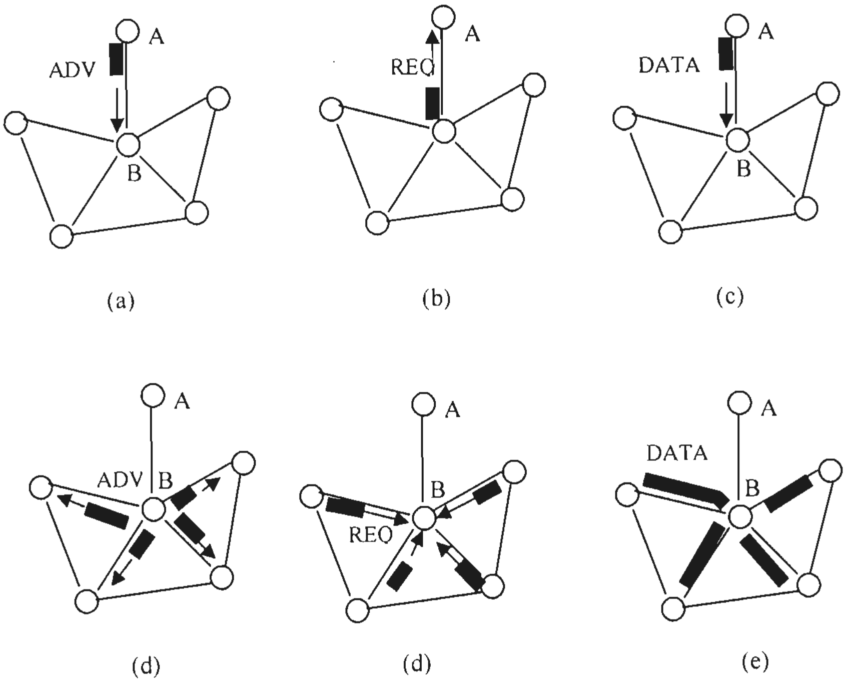
Figure 2.3. Directed Diffusion interest propagation and final path set

routes from multiple sources to a single destination that allows in-network consolidation of redundant data.

Sensors measure events and create gradients of information in their respective neighborhoods. The sink requests data by broadcasting interests. An interest describes a task required as above and is diffused through the network hop by hop. Each node on the way broadcast it to its neighbors. As the interest is propagated throughout the network, gradients are set up to draw data satisfying the query toward the requesting node. Each sensor that receives the interest sets up a gradient towards the SNs from which it receives the interest. This process continues until gradients are set up from the sources back to the sink. In simplest form, a gradient specifies an attribute value and a direction. The strength of the gradient may be different toward different neighbors, resulting in different amounts of information flow. At this stage, loops are not checked, but are removed at a later stage. When interests fit gradients, paths of information flow are formed from multiple paths, and then the best paths are reinforced to prevent further flooding.

Figure 2.3 shows initial interest propagation, gradients setup and final reinforced path. Each node receiving the interest can do caching for later use. The nodes also have the ability to do in-network data aggregation, which is modeled as a minimum Steiner tree problem [16]. The interests in the caches are then used to compare the received data with the values in the interests.

Sensor Protocols for Information via Negotiation (SPIN) [143] [65]: SPIN is one of the earliest data-centric routing mechanisms for WSN routing. SPIN disseminates all the information at each node to every node in the network assuming that all nodes in the network are potential sinks. This enables a user to query any node and get the required information immediately. These protocols make use of the property that nodes in close



(a) (b) (c)

(d) (d) (e)

Figure 2.4. Different stages of metadata exchange in SPIN

proximity have similar data, and hence there is a need to only distribute the data other nodes do not posses.

The idea behind SPIN is that before transmissions, high-level descriptors or meta-data are exchanged. Each node upon receiving new data, advertises it to its neighbors and interested neighbors (i.e. those who do not have the data) retrieve the data by sending a request message. This exchange of meta-data among nodes solves many problems of classical flooding such as redundant information passing, overlapping of sensing areas and resource blindness which results in lot of energy savings. There are three messages defined in SPIN to exchange data between nodes as:

- *ADV* (a new data advertisement): When a SPIN node has data to share, it can advertise this fact by transmitting an ADV message containing meta-data.
- *REQ* (request for data): A SPIN node sends an REQ message when it wishes to receive some actual data.
- *DATA* (data message): DATA messages contain actual sensor data with a meta-data header.

Figure 2.4 summarizes the steps of the basic SPIN protocol as: Node A starts by advertising its data to node B. Node B responds by sending a request to node A. After receiving the requested data from A, node B then sends out advertisements to its neighbors, who in turn send requests back to B. B sends data to nodes who sent request messages.

However, SPIN family of protocols comprises following protocols specifically designed for different scenarios:

SPIN-PP: This protocol has been designed to perform optimally for point-to-point communication. This protocol is a simple 3-way handshake protocol in which energy is not considered to be a constraint. One major advantage of using this protocol is its simplicity and that each node needs to know only about its single-hop neighbors and does not require any other topology information.

SPIN-EC: In this protocol, the SNs communicate using the same 3-way handshake protocol as in SPIN-PP but there is an energy-conservation heuristic added to it. A node will participate actively in the protocol only if it is above a certain energy threshold and believes that it can complete all the other stages of the protocol.

SPIN-BC: This protocol was designed for broadcast networks in which the nodes use a single shared channel to communicate. Nodes within a certain range of the sender receives message. Node which receives an ADV message does not immediately respond with an REQ message. It waits for a certain time before sending out the REQ message. When a node other than the advertising node receives the REQ message, it cancels its own request so that there are no redundant requests for the same message. When the advertising node receives an REQ message, it sends the data message only once because it is a broadcast network even though it might have got multiple requests for the same message.

SPIN-RL: This protocol differs from SPIN-BC by making two changes. Each node remembers all the advertisements and the ids of the nodes it hears from. If within a certain time period it does not receive any requested data, it resends request. Nodes have a limit on the frequency with which they resend the data messages. After sending a data message, a node waits for some time before responding to other requests for the same data message.

Rumor Routing [25]: Rumor routing is a Directed Diffusion variant and is mainly intended for contexts in which geographic routing criteria is not applicable. Generally Directed Diffusion floods the query to the entire network when there is no geographic criterion to diffuse tasks. In many scenarios, data requested is too less as compared to queries issued and thus use of query flooding is unnecessary, rather it is better to flood events. Rumor routing is between event flooding and query flooding. Rather than flooding the entire network to retrieve information, here queries are routed to the nodes that have observed a particular event.

In order to flood events through the network, the rumor routing algorithm employs long-lived packets, called agents. When a node detects an event, it adds such event to its local table and generates an agent. Agents travel the network to spread information about local events to distant nodes. Once a node generates a query for an event, nodes that know the route respond to the query by referring its event table and flooding of whole network is avoided. Unlike DD where data can be sent through multiple paths at low rates, rumor routing maintains only one path between source and destination.

Low-Energy Adaptive Clustering Hierarchy (LEACH)[148]: This has already been explained in Section 2.2 since it uses clustering as well as routing. LEACH is a self-organizing and adaptive clustering protocol using randomization to evenly distribute the energy expenditure among the sensors. Clustered structures are exploited to perform data aggregation where cluster heads act as aggregation points.

Gradient Based Routing (GBR) [18]: GBR is another variant of DD that memorizes the number of hops when an interest is diffused through the whole network. Each node calculates height of the node, which is the minimum number of hops to reach the sink. A link with the largest gradient is selected to forward packet where gradient is the difference between a node's height and that of its neighbor. GBR uses data aggregation and traffic spreading in order to uniformly divide the traffic over the network.

Threshold sensitive Energy Efficient sensor Network protocol (TEEN) [4]: Manjeshwar and Agarwal propose TEEN, a data-centric protocol designed for networks where response time is expected to be too low i.e. it reacts to sudden changes in the sensed attributes (reactive networks). As shown in Figure 2.5, TEEN architecture is hierarchical where closer nodes form clusters with one of node taking up the responsibility of cluster head. Cluster heads further form second level clusters and this process goes on until sink is reached. The main features of such architecture are that all the nodes need to transmit only to their immediate cluster-head and only the cluster head performs additional computations on the data which saves energy. Cluster-heads at higher levels in the hierarchy transmit data over correspondingly larger distances and perform extra computations, hence consume more energy. In order to evenly distribute this consumption, all nodes take turns becoming the cluster head for a time interval T, called the cluster period.

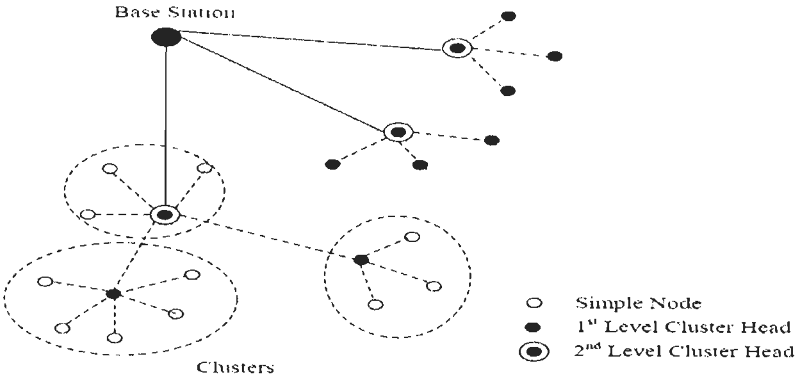


Figure 2.5. TEEN and APTEEN

Adaptive Threshold sensitive Energy Efficient sensor Network protocol (APTEEN) [5]: Manjeshwar and Agarwal further propose APTEEN, a hybrid protocol that changes the periodicity or threshold values used in the TEEN protocol according to user needs and the application type. APTEEN is an extension to TEEN and uses same architecture as TEEN aimed at both capturing periodic data collections and reacting to time-critical events. When sink forms clusters, the cluster heads broadcast attributes (a set of physical attributes describing required information), the threshold values, the transmission schedule to all nodes (a TDMA schedule, assigning a slot to each node), and count time (the maximum time period between two successive reports sent by a node). Cluster heads also perform data aggregation in order to save energy. APTEEN supports three different query types:

- historical, to analyze past data values
- one-time, to take a snapshot view of the network and
- persistent to monitor an event for a period of time

Power-Efficient GAthering in Sensor Information System (PEGASIS) [120]: PEGASIS is an improvement of the LEACH (C121) protocol. Key idea is to organize SNs in a chain where nodes take turns to act as chain leader and thus energy expenditure is distributed among the nodes in the network. Either sink builds chain (centralized) or a greedy algorithm is run at each node in a distributed manner. However, in either case construction of the chain requires global knowledge of the network at all nodes.

Node furthest from the sink starts chain building process. Closest neighbor to this node is chosen as the next one in the chain, and so on. TDMA base time schedule is used to select chain leader in such a manner that during each transmission round it is at a different position in the chain. Each node receives data from a neighbor and aggregates it with its own reading by generating a single packet of the same length. This aggregated packet is transmitted to the next node in the chain until the packet reaches the current chain leader. The leader includes its own data into the packet and sends it to the sink. Each node is assumed to communicate with the sink directly.

Hierarchical-PEGASIS [121]: Hierarchical PEGASIS is an extension to PEGASIS, with the objective to reduce packet transmission delay to sink. To accomplish this, simultaneous transmissions of data are studied in order to avoid collisions through approaches that incorporate signal coding and spatial transmissions. Only spatially separated nodes are allowed to transmit at the same time. The chain-based protocol with CDMA-capable nodes constructs a chain of nodes that forms a tree-like hierarchy, and each selected node at a particular level transmits data to a node in the upper level of the hierarchy. This ensures parallel data transmission and reduces delay. Hierarchical PEGASIS perform better than the regular PEGASIS scheme by a factor of about 60.

Constrained anisotropic diffusion routing (CADR) [85]: Chu *et al.* propose CADR to come up with a routing paradigm which is general form of DD. To select from which sensors to get the data, it diffuses queries by using a set of information criteria. This is achieved by selectively activating only the sensors that are close to an event and dynamically adjusting data routes. The key idea is to maximize information gain and to minimize latency and bandwidth.

While calculating gradients, DD considers only communication cost whereas CADR also considers information gain in addition to communication cost. Each node evaluates an information/cost objective and routes data based on so computed local information/cost gradient and end-user requirements.

COUGAR [164] [163]: Yao and Gehrke propose COUGAR, an approach which views WSN as a huge distributed database system. The key idea is to use declarative queries in order to hide physical details of network from the user i.e. query processing is abstracted from the network layer functions such as selection of relevant sensors and so on. COUGAR utilizes in-network data aggregation to obtain more energy savings. It uses a loosely-coupled distributed architecture to support both aggregation and more complicated in-network computation. In its architecture, there is a new query proxy layer on each SN, interacting with both routing layer and application layer. SNs select a leader node to perform aggregation and transmit the data to the sink. The sink is responsible for generating a query plan that specifies the necessary information about the data flow and in-network computation for the incoming query, and sends it to the relevant nodes.

Sequential Assignment Routing (SAR) [76]: Sequential Assignment Routing (SAR) is routing scheme which concentrates on the energy efficiency and QoS factors. It creates multiple paths from the nodes to the sink each rooted from the nearest neighbours of the sink node and helps in achieving a more energy efficient structure as well as maximizes the fault tolerance of the network. Each node tries to increase the tree or extend its roots by adding all those other neighbour nodes connecting the sink node. Nodes in the network which are low in energy reserves and which do not support real time factors like redundancy, bounded latency are deleted or ignored to be added as paths towards the sink. When a tree construction is completed, every node has multiple paths from it through other nodes to reach some other node or to reach the sink. Using this structure, every node is capable of transmitting to all the other nearest single hop neighbours.

2.4.2 Routing without In-network Data Aggregation

Minimum Cost Forwarding Algorithm (MCFA) [42]: Fan Ye *et al.* propose a Minimum Cost Forwarding Algorithm (MCFA) intended for scenarios where SN need not have a unique ID nor maintain a routing table. It exploits the fact that the direction of routing is always known (i.e., toward the fixed external sink). Each node maintains the least cost estimate from itself to the sink. SN broadcasts each forwarding message to its neighbors. When a node receives the message, it checks if it is on the least cost path between the source SN and the sink. If this is the case, it rebroadcasts the message to its neighbors. This process repeats until the sink is reached.

Least cost path estimate from node to the sink is calculated as follows: The sink broadcasts a zero cost, while every node initially sets its least cost to the sink to infinity. Each node, upon receiving the broadcast message originated at the sink, checks to see if the estimate in the message plus the link on which it is received is less than the current estimate. If yes, the current estimate and the estimate in the broadcast message are updated. If the received broadcast message is updated, it is resent; otherwise, nothing further is done.

Energy Aware Routing (EAR) [116]: Shah and Rabaey propose energy aware routing mainly intended to enhance network survivability which is the main metric for proposed approach. The approach argues that using the minimum energy path all the time will deplete the energy of nodes on that path. EAR occasionally use a set of sub-optimal paths to increase the lifetime of the network. Sub-optimality is computed by means of a probability function, which depends on the energy consumption of each path. One of the multiple paths is used with a certain probability so that the whole network lifetime increases. The protocol assumes that each node is addressable through a class-based addressing which includes the location and types of the nodes.

All routes between a source-destination pair and their costs are discovered using localized flooding. Based on discovered route and cost, a forwarding table is built by choosing neighboring nodes in a manner that is proportional to their cost. Accordingly, forwarding tables are used to send data to the destination with a probability inversely proportional to the node cost. Localized flooding is performed by the destination node to keep the paths alive.

Minimum energy communication network (MECN) [141]: Rodoplu and Ming propose a protocol which computes an energy-efficient sub-network and the minimum energy communication network (MECN), for a certain sensor network utilizing low-power GPS. MECN identifies a relay region for every node. This region consists of nodes in a surrounding area where transmitting through those nodes is more energy-efficient than direct transmission. The enclosure of a node *i* is created by taking the union of all relay regions node *i* can reach. The main idea of MECN is to find a sub-network that will have fewer nodes and require less power for transmission between any two particular nodes. Each node performs localized search in its relay region to find global minimum power paths without considering all the nodes in the network. MECN is self-reconfiguring and thus can dynamically adapt to node failure or the deployment of new sensors.

Small minimum energy communication network (SMECN) [81]: SMECN is an extension to MECN. MECN assumes that every node can transmit to every other node, which is not possible every time. Hence, SMECN is designed to overcome possible obstacles between any pair of nodes. However, alike MECN the network is still assumed to be fully connected. The subnetwork constructed by SMECN for minimum energy relaying is provably smaller (in terms of number of edges) than in MECN. Hence, the subnetwork constructed by SMECN is smaller than the one constructed by MECN if the broadcast region is circular around the broadcasting node for a given power setting.

Two-tier Data Dissemination (TTDD) [53]: Haiyun *et al.* propose TTDD, a grid based approach that provides scalable and efficient data delivery to multiple, mobile sinks. Each data source in TTDD proactively constructs a virtual grid structure over entire network and enables mobile sinks to continuously receive data on the move by flooding queries within a local cell only. Each cell of a grid acts as one cluster since a sink in this cell sends queries and receives data (single or multi-hop) through one of grid corner nodes as defined next.

In TTDD, SNs are assumed stationary and location-aware, whereas sinks may slowly change their locations. Once an event occurs, nodes that have event in their sensing range (called active nodes) process the signal, and one of them becomes the source to generate data reports and acts as starting point for grid construction. To build grid structure, data source sends a data announcement message to each of its four adjacent crossing points using simple greedy geographical forwarding. When message reaches a node closest to a crossing point, it stops and that node is treated as representative node for that crossing point and is called as dissemination node. During this process, each intermediate node stores the source information and further forwards the message to its adjacent crossing points except the one from which the message comes. This process continues until the message stops at the boundary of the sensor field. In this way, a virtual grid structure with square sized cells is obtained.

Using this grid, a sink can flood a query, which will be forwarded to the nearest dissemination point in the local cell to receive data. Then the query is forwarded along other dissemination points upstream to the source and in response data flows down in the reverse path to the sink. Trajectory forwarding is used to handle sink movement in the sensor field. The path length for query/data in TTDD is not shortest path, yet authors believe that the sub-optimality in the path length is worth the gain in scalability.

The major drawback of TTDD is that every time a new event appears in sensor field, a new grid is constructed. This consumes significant amount of network energy and costs major network overheads if number of events are more. Also, if application has hard real-time constraints, it does not provide optimal packet delay due to sub-optimal paths.

Geographic adaptive fidelity (GAF) [162]: GAF is primarily proposed for wireless ad hoc networks, but may be applicable to WSNs as well. This is an energy-aware location-based routing algorithm which conserves energy by turning off unnecessary nodes in the network without affecting the level of routing fidelity. It forms a virtual grid over entire sensor field and each node uses its GPS-indicated location to associate itself with a point in the virtual grid. Nodes associated with the same point on the grid are considered equivalent in terms of the cost of packet routing. This helps to keep certain nodes in particular grid area in sleeping state and thus saves energy. Nodes change states from sleeping to active in turn so that the load is balanced. GAF substantially increases the network lifetime as the number of nodes increases.

Geographic and Energy Aware Routing (GEAR) [165]: Yu *et al.* propose Geographic and Energy Aware Routing (GEAR) that uses energy-aware and geographically informed neighbor selection heuristics to route a packet toward the destination region. The process of forwarding a packet to all the nodes in the target region consists of two phases:

- Forwarding the packets towards the target region: For packet forwarding there are two cases to consider. First, when a closer neighbor to the destination exists, GEAR picks a next-hop node among all neighbors that are closer to the destination. Second, when all neighbors are further away (i.e. there is a hole), GEAR picks a next-hop node that minimizes some cost value of this neighbor.
- Disseminating the packet within the region: Under most conditions, GEAR uses a Recursive Geographic Forwarding algorithm to disseminate the packet within the region. However, under some low density conditions, recursive geographic forwarding sometimes does not terminate, routing uselessly around an empty target region before the packet's hop-count exceeds some bound. In these cases, restricted flooding is used.

Thus, the key idea is to restrict the number of interests in directed diffusion by only considering a certain region rather than sending the interests to the whole network. By doing this, GEAR can conserve more energy than directed diffusion.

SPEED [129]: SPEED is a routing protocol that ensures soft real-time end-to-end guarantees to provide QoS for WSNs. It requires each node to maintain information about its neighbors and uses geographic forwarding to find the paths. Also, SPEED tries to ensure a certain speed for each packet in the network so that each application can estimate the end-to-end delay for the packets. Before making the admission decision, end-to-end delay is computed as the distance to the sink divided by the speed of the packet. Moreover, SPEED can provide congestion avoidance when the network is congested.

Stojmenovic and Lin [60] propose Most Forward within Radius (MFR), Geographic Distance Routing (GEDIR), and DIR (a compass routing method). These protocols are position aware and deal with basic distance, progress, and direction-based methods. Another position-based algorithm called SPAN [14] selects some nodes as coordinators based on their positions. SPAN builds on the observation that when a region

Table 2.2. Comparison of Routing Protocols for WSNs

Category	Routing Protocol/ Technique	Network Topology	Geographical Scoping	Scalability	Data Retrieval
Routing With Aggregation	Directed Diffusion	Flat	No	Limited	Query Driven
	SPIN	Flat	No	Limited	Query Driven
	Rumor Routing	Flat	No	Good	Query Driven
	LEACH	Hierarchical	No	Good	Event Driven
	GBR	Flat	No	Limited	Query Driven
	TEEN & APTEEN	Hierarchical	No	Good	Event Driven
	PEGASIS/ Hierarchical-PEGASIS	Flat/ Hierarchical	No	Good	Event Driven
	CADR	Flat	No	Limited	Event Driven
	COUGAR	Flat	No	Limited	Query Driven
	SAR	Flat	No	Limited	Query Driven
Routing Without Aggregation	MCFA	Flat	No	Good	Event Driven
	EAR	Flat	Yes	Limited	Query Driven
	MECN & SMECN	Flat	Yes	Low	Event Driven
	TTDD	Hierarchical	No	Low	Both Possible
	GAF	Hierarchical	Yes	Good	Event Driven
	GEAR	Flat	Yes	Limited	Event Driven
	SPEED	Flat	Yes	Limited	Query Driven
	SPAN	Flat	No	Limited	Event Driven

of a shared channel wireless network has a sufficient density of nodes, only a small number of them need be on at any time to forward traffic for active connections. SPAN is a distributed, randomized algorithm where nodes make local decisions on whether to sleep, or to join a forwarding backbone as a coordinator. Each node bases its decision on an estimate of how many of its neighbors will benefit from it being awake and the amount of energy available to it. The coordinators form a network backbone used to forward messages. A node should become a coordinator if two neighbors of a non-coordinator node cannot reach each other directly or via one or two coordinators.

2.4.3 Comparison of Routing Protocols

Table 2.2 compares above routing techniques based on following parameters:

Network Topology: It defines whether all nodes are treated at one level only (i.e. flat) or multiple levels (i.e. hierarchical). In hierarchical topology nodes are arranged in groups/clusters where one of the nodes acts as coordinator/head for group/cluster members and does all communications further with higher level node or sink.

Geographical Scoping: Routing schemes which utilize the position information to relay the data to the desired regions rather than the whole network are known to use geographical scoped query and data.

Scalability: Scalability defines capability of a routing technique to give satisfactory throughput under an increased load when more nodes are added and sensing area is extended.

Data Retrieval: It defines whether data is disseminated to sink is query driven or data driven. In query driven data flows only in response to a query from sink. Whereas, in event driven, data flow is triggered by generation of new data as a result of appearance of event or change in the property of event or in response to continuous monitoring after every dissemination interval.

2.5 Caching

Caching has since long been used in database systems and operating systems to enhance data availability, to reduce data retrieval time and to reduce network overheads by making data available very close to the data requester. With the emergence of Internet, caching has also found immense focus and use in web. Caching in web is used both in a cooperative [49] [78] (i.e. when storage from different machines and/or locations cooperate to give one larger cache abstraction) and non-cooperative environments [27] [10] [31]. Michael *et al.* [94] propose optimal or near-optimal replacement policies for web objects. The optimization problem is formulated as a Markov decision process. Cellular networks which have nodes with higher hardware/software resources than WSNs and more powerful data sources, have also benefited from caching by replicating copies of data near to the users [28].

2.5.1 Caching in Mobile Ad Hoc Networks

Caching and replication find natural place in mobile ad hoc networks (MANETs). By exploiting the priori knowledge of the network, various data replication schemes have been proposed/used to meet channel access demands [128] [126]. Caching schemes such as [41] [90] [79] are used for MANETS which do not necessarily require priori knowledge of network. N. Chand *et al.* propose broadcast based cache invalidation and prefetching for Mobile Environment [96] and also propose energy efficient cache invalidation in a disconnected wireless mobile environment [98]. Manish and Barua [82] incorporate caching of routes in AODV protocol used in MANETs with the aim to reduce the routing and MAC load of AODV without changing the basic structure of the protocol

The most relevant research works to our protocol are the cooperative caching protocols which have been developed for MANETs and WSNs. The main motive for the development of these protocols for MANETs is to make data available despite mobility of the nodes. Cooperation among various nodes have greatly been exploited to give better performance in terms of high data availability, low data latency, reduced congestion on the channel and reduced energy consumption. Das *et al.* [119] propose a cooperative caching scheme for IMANETs. A broadcast based simple search scheme is proposed to establish cooperation among all mobile hosts (MHs) in the network to share cached data items. Several cache replacement strategies were proposed in the context of wireless cooperative caching. Hara *et al.* [127] presented a series of cooperative caching schemes (LOP, GOP and SOP) for push-based information system. The aim of these strategies is to shorten the average response time for data access by replacing cached items based on their access frequencies, the network topology, and the time until the next broadcast. Later, Hara proposed schemes to deal with data updates also.

Lau *et al.* [149] propose a cooperative caching architecture for supporting continuous media proxy caching. They introduced an application manager to transparently perform data location and session migration of continuous media streams among all proxy caches. Sailhan and Issarny [40] propose a cooperative caching based on Zone Routing Protocol which increases data accessibility by P2P communication among mobile nodes when they are out of bound of a fixed infrastructure. Nuggehalli *et al.* [106] addresses the problem of optimal cache placement in ad hoc wireless networks and proposed a greedy algorithm, called POACH, to minimize the weighted sum of energy expenditure and access delay. Shen *et al.* [54] propose a broadcast based cooperative caching scheme for hybrid networks where a client shares the caches of clients lying in its proximity. Bandwidth and energy consumption to locate a client having cached the requested data is very high due to flooding of the messages.

Chand *et al.* [97] proposed a cooperative caching strategy that partitions network into non-overlapping clusters based on the physical network proximity. For a local cache miss, each client looks for data item in the cluster. If no client inside the cluster has cached the requested item, the request is forwarded to the next client on the routing path towards server. Yin *et al.* [80] proposed hybrid cooperative caching for ad hoc networks where both data and data path are cached at some mobile nodes. Hybrid approach avoids weaknesses of both when only data is cached (CachData) and when only data path is cached (CachePath). GroCoca [23] exploits data request and node mobility patterns to organize nodes into groups. In this approach, peers exhibiting similar mobility patterns and displaying similar data affinity form tightly-coupled group (TCG). Two cooperative cache management protocols, namely, cooperative cache admission control and replacement, are designed to control data replicas and improve data accessibility in TCGs.

2.5.2 Caching in Wireless Sensor Networks

Due to very limited memory and large number of SNs, caching in WSN is still a very challenging task. However, with advancements in MEMS and microelectronics the SN resource constraints are relaxing and enough memory is now available at SN to be used as small cache. Some efforts are already made to introduce caching for WSNs. Dunkels *et al.* [24] introduce a mechanism called Distributed TCP Caching (DTC). The DTC mechanism uses segment caching and local retransmissions to avoid expensive end-to-end retransmissions. Shashi *et al.* [73] proposed a Steiner tree based approach to find locations in the network to cache data that minimize packet transmissions and reduces the power consumption in the network, and hence extends its lifetime. Finding locations of the nodes for caching data to minimize communication cost corresponds to finding the nodes of a weighted Minimum Steiner tree whose edge weights depend on the edge's Euclidean length and its data traffic rate. Rahman and Sajid [89] proposed certain ways to improve energy efficiency in a WSN that already use some routing technique by emulating cache where latest sensed data are always known without requiring the sensors to sense and report continuously. They exploit data negotiation, data change expectancy, and data vanishing for a given scenario. Dimokas Nikos *et al.* [30] proposed cooperative caching solution particularly suitable for Wireless Multimedia Sensor Networks (WMSNs). This approach estimates the importance of SNs relative to the network topology and selects few as coordinators for the caching decisions to provide information about accessing the requested data or even as caching points.

Approaches proposed by Shashi *et al.*, and Dimokas *et al.* try to find nodes in the network which can provide suitable locations for caching data. Suitability can be defined by considering performance metrics like data delivery time and energy consumption etc. Once these nodes are found, all queries and data generated must flow towards these nodes and hence defines query/data dissemination route.

2.6 Multi-Radio Modes of a Sensor Node

Fast changing application domain of WSNs demand various operations at SN level, few of which require very high bandwidth (also transmission range) and computational power, where as others require low bandwidth and low processing power. Also, radio transmission range requirement is different for different operations and different scenarios. Therefore, dual-radio, dual-processor nodes are fast emerging that provide low energy operation as well as substantially increased computational performance and communication bandwidth. In such systems, secondary radio and processor works with much less power than the main radio (IEEE 802.11) and processor. Accordingly, secondary (low power) radio transmits to small distance as compared to high power radio.

A lot of work has already been initiated to develop and exploit multi mode capabilities in SNs. Stathopoulos *et al.* [132] propose a topology control mechanism for establishing end-to-end paths in networks of dual-radio nodes, where the secondary radios are used as a multihop control channel. The topology control mechanism selectively wakes up only the nodes required for the path to form, by sending control messages to them through the low-bandwidth radio. Authors claim to achieve energy savings of more than 60% compared to a mechanism that wakes up all the nodes while incurring up to 12 seconds of additional delay. Some times in heterogeneous environments, 32-bit CPU nodes must communicate with 8 or 16 bit microcontroller based nodes. Hence, new generation of 32-bit nodes, as for example the LEAP node [29] includes an on-board low-power microcontroller (for constantly vigilant operation) and a second, low-bandwidth radio.

Work in [38] and [91] exploit higher capacity (transmission range/bandwidth) of different radios present on nodes. Instead of using high bandwidth (and high power) radio, Shih *et al.* [37] and Jun *et al.* [52] save energy by exploiting low-bandwidth radio for resource discovery. [130] propose to use low bandwidth radio as a control channel to perform networking functions such as access point association and [19] use it for scheduling transmissions. [12] introduces a technique of Dual Channel Multiple Access with Adaptive Preamble (DCMA/AP). The protocol uses two separate frequencies for data and control packets to avoid the use of handshake mechanisms (e.g. RTS/CTS) in order to reduce energy consumption and packet delay. Paramvir *et al.* [101] presented a new design that includes multiple radios that work together to accomplish a common task. By using multiple radios, each of which does different things well, and by integrating them at the systems level, they argue that it provide improved performance and greater functionality to the users. Yuvraj *et al.* [159] show importance of using Bluetooth radios instead of 802.11b standard operating modes to serve as a paging channel for the 802.11b wireless LAN.

As pointed out in [132], in terms of energy per bit transmitted, high-bandwidth radio operates with much greater energy efficiency than the low bandwidth radio. For example, 802.11g consumes 112 nJ/bit as opposed to 979 nJ/bit for 802.15.4. However, if amount of data to be sent is very small or occasional data is transmitted or no data is to be transmitted at all (i.e. some local activity at node is required), the much higher state transition cost and idle energy consumption of high-bandwidth radio (more than 10 times than low-bandwidth radio) proves counter-productive to use it. Instead, in order to reduce energy consumption, the high-bandwidth radio should be kept off, to be activated only when there is a significant amount of data that needs to be transmitted. Whereas, low-bandwidth radio is less energy efficient but consumes much less energy when idle. It can quickly make transition from sleep to active state, send the necessary data, and then deactivate without consuming much energy. Thus low-power low-bandwidth radio is ideal if small chunks of data need to be transmitted occasionally as well as remaining "vigilant" for long time periods.

Finally, the discussion on clustering, routing and communication in WSN can never be complete without a word on 'security (network/information)', on of the inherent problem with any kind of network today. Further, in case of wireless networks like ad hoc WSNs security problem becomes more severe. Communication between a node and its cluster head in WSN is wireless and broadcast based. The security threats can stem from numerous reasons like information in the air can be captured by the adversary and decrypted; a node from the network can be captured and information read; a false node can be added to the network/cluster pushing wrong data into the network or blocking the authentic data. Hence main security issues are: key establishment, secrecy, authentication, privacy, robustness to denial-of-service attacks, secure routing, and node capture. Perrig *et al* [7] propose a protocol TESLA (Timed, Efficient, Streaming, Loss-tolerant Authentication Protocol) for source authentication, which provides confidentiality and authentication using only RC5 shared key decryption primitive. Perrig *et al* [8] propose SPINS (Security Protocols for Sensor Networks) comprised of Sensor Network Encryption Protocol (SNEP) and μTESLA (the "micro" version of the Timed, Efficient, Streaming, Loss-tolerant Authentication Protocol). The function of SNEP is to provide confidentiality (privacy), two party data authentication, integrity and freshness. μTESLA is to provide authentication to data broadcasts. Ensuring secure and authentic group communication where members of a group/cluster only hear to communication with in group/cluster and also ensuring authentic joins and leaves by a node to group/cluster is very vital. Some important work to solve these problems is given in [142] [66] [111]. Also, defending wireless channel in a WSN from intentional interference and channel jamming causing denial-of-service kind of situation is important to ensure uninterrupted monitoring and control. The most recent work in the area is given in [144] [145] [105].

2.7 Shortcomings and Research Gaps

Following are the shortcomings and research gaps that have been observed:

1. A lot of work in WSN study exists to reduce energy concumption in a WSN. Few of them are: to compress data before dissemination; to find optimal paths between source and sink for data transfer; to find proper aggregation points in the network; aggregate smaller units into larger units (fusion) for transmission from different sources or generated at different times. For example, work in [122] [166] [107] talks about exploiting correlation to decide when to do aggregation and how to forward highly correlated data as well as unrelated data. However, insignificant work yet exists to block and isolate undesired/unnecessary data nearer to its source of generation based on its value. Blocking such data early in its journey towards sink can avoid a lot of unnecessary inter-node transmissions resulting in energy savings.
2. Almost entire work in the literature reveal that sensing frequency set at active SNs (that sense an event) is one time activity and is set only once prior to the start of the network operation which remains same for entire sensing period. Since event behaves unpredictably most of the time, to adjudge best sensing frequency before network operation is extremely difficult. Even if some how near optimal sensing frequency is found, due to random behaviour of event energy spent in unnecessary sensing (when event has not changed state) can not be avoided. However, o conserve network energy a lot of schemes do exist that shuffle the state of SNs among active, sleep and awake etc based upon event proximity from these nodes, but no scheme has been found that dynamically changes the sensing frequency at active SNs. Therefore, we have a strong intuition that if sensing frequency at active SNs can somehow be made to respond to event behaviour and set accordingly on the fly, a lot of energy can be saved in

unnecessary sensing and subsequently number of inter-node transmissions can be reduced.

3. The use of multi-radio nodes is very recent trend. As described in previous section, among others work like [132] [37] [52] do use dual-radio nodes where low power radio is used either for topology control or resource discovery. No work yet exists where paths are decided/established based on different transmission ranges of dual-radios. Paths are established either considering nodes in entire sensor field at one level (i.e. flat) or by exploiting certain hierarchies created based on clusters, groups or virtual grids. Till now, all path setup schemes, clustering and virtual grid formation schemes as discussed previously use single transmission/reception range of a primary radio. However, we foresee that if dual radio ranges (i.e. high power and low power) of a dual-radio device present on a node are used to decide cluster size, cell size of a virtual grid etc, more energy efficient paths can be found.
4. With new emerging application scenarios of WSNs, there is need to temporarily hold data in the network before it becomes stale or is consumed by sink. Need of temporary storage arises from the fact that either sink does not need data at present but may require after some time, or being one of SNs having limited memory is currently swamped by excess data flow or sink need to serve a query received from out side that needs information which can only be inferred from data collected from within a small time period. Work in literature [73] [30] do exist which tries to find aggregation points in network, finds better points for temporary storage and even allocates nodes that collaborate to accomplish the task of temporary storage. But, a complete scheme that handles excess data flow from source nodes, caches it for subsequent retrieval by sink and maintains consistency and coherency among various data items is almost absent. A lot of work is required in this direction to handle voluminous data energy efficiently within the WSN.

2.8 Summary

This chapter discusses all components of a data dissemination paradigm namely clustering, in-network processing (especially data aggregation), routing and caching. Also, the use and emergence of multimode SNs is discussed at the end and few words on security in WSNs concludes the discussion. At the end, shortcomings and research gaps in existing work that prompted us to take present study have been given. Constrained resources especially limited battery life of a node and unique nature of WSN pose great challenge to designers of WSN at every layer of networking. Throughout the literature review, one can easily observe that most of the prior work in WSN design revolve around enhancing network lifetime by reducing node energy depletion using efficient clustering, optimal routing and by applying certain in-network processing on data. Still, the objective to energy efficiently disseminate data from sources to sink(s) is yet far from being achieved. As new WSN application scenarios emerge, new design constraints surface.

Chapter 3

Two-Way Sliding Window (TWSW) Data Filtering

3.1. Introduction

Ad hoc deployment at inhospitable locations, auto-reconfiguration, self healing and unattended nature make WSNs scalable and very powerful tool for applications like habitat monitoring, process monitoring and control, environmental monitoring, traffic monitoring, infrastructure protection, remote monitoring and control, military surveillance and command etc [26] [63] Most of these applications involve continuous sensing and data dissemination activities that accumulate huge amount of data generated at SNs over very long periods of time. This consumes significant amount of nodes' scarce energy. Over the past several years, several methods to reduce energy consumption in WSNs have been explored by proposing many energy efficient routing protocols [68], various in-network data processing strategies [36] to reduce the bulk of data to be transmitted, new radio propagation models, many sensor data compression schemes [72] and various optimizations at hardware and software design levels. However, due to diverse application scenarios and unique nature of WSN, the challenge still remains far from being solved.

Since, communication is the major source of energy consumption in WSNs [45], number of transmissions among various nodes of WSN must be minimized. Most of the proposed routing protocols are based on reducing the path traversed by the data and queries by finding optimal paths between set of source nodes and set of sinks or by using caching at intermediate nodes or by using sink(s) movements for collecting data etc. Apart from these routing protocols various in-network data processing strategies including data aggregation have been proposed which process data and/or queries at some nodes within the network on its way to destination. Depending upon the application requirements data aggregation (also termed as data fusion) may employ lossy or lossless data fusion or data compression. This reduces the amount of data to be transferred to sink(s) and hence reduces inter-node transmissions. While processing data in-network, one must be careful not to defeat collaborative effort of SNs by applying some fusion or compression which diminishes granularity of sensed data below a particular level required at sink. Another major concern while performing in-network processing is the delay introduced due to processing time. If network is designed to handle real time applications like process control or multimedia applications, delay can significantly affect performance and make data useless/incorrect.

Monitoring vast area using randomly distributed SNs over a long period results in generation of huge amount of data. Due to unpredictable event behaviour the data captured at SNs may have few redundant and/or spurious data readings. In most such scenarios, event changes relatively slowly during major portion of an observation period but also has a possibility of sudden changes in its behaviour during certain small unpredictable periods in between. If these abrupt fluctuations in event behaviour cause sensed readings still with in maximum/minimum expected limit, they carry useful information and are not spurious. Also, sensing frequency at SNs is set one time prior to the start of network operation and is not altered during entire observation period. Hence, for SNs not to miss readings about event parameter at crucial times and to capture even minute variations in event behaviour, there is no other option than to set sensing frequency very high at SNs. High sensing frequency results in generation of very large number of data reports per unit time. As event is relatively static during most of the time, only few of these readings reveal useful information to application running at sink whereas others are redundant. If successive readings/reports received at sink do not vary at all from each other or vary insignificantly (not differentiable to application at sink), they are redundant. For example, if application's data granularity limit is 0.05 °C and last temperature reading received has a value as 20°C, then successive reading with any of values 19.97°C, 20.025°C, 20.04°C etc is a redundant reading.

However, unlike above discussed case if difference between currently sensed reading and previously disseminated reading is beyond maximum/minimum expected limit, reading is spurious. This type of spurious data may be generated in sensor field due to various reasons like temporary abnormal conditions prevailing around few active SNs (for example, temporary obstruction caused by external object in the way between event and SN), malicious data injection by an adversary or due to malfunctioning of hardware/software at SNs etc. For a given sensing scenario, this maximum/minimum limit can be set suitably by general heuristics using existing data sheet (similar if not same) or following patterns from various temperature data sources. For example, suppose a WSN is deployed for monitoring vast agricultural area for temperature variation due to sun and data reporting time to sink is 15 minutes. Maximum/minimum limit on variation in sensed reading within above reporting time can loosely be set by following historical temperature sheet available and say here it is set as 2°C i.e. field temperature can not vary more than 2°C in 15 minutes even if it is peak hot/cool hour of the day. Hence, if last temperature reading received by sink has a value as 20°C, then successive reading with value 22.5°C or 15°C is treated as spurious, whereas readings with value 21.7°C or 19.2°C are non-spurious.

Redundant and spurious data readings reduce the quality of data and undesirably consume scarce network energy. Therefore, we predict huge gains in terms of network energy savings and network overhead reduction if one can employ some in-network mechanism to suppress or filter out these undesired data nearer to source so as to avoid their injection further towards sink. This will certainly reduce large number of inter-node transmissions leading to precious network energy savings.

Hence, in this chapter we propose a scheme to suppress/filter undesired data which is based on some data filtering/blocking at certain SNs and is known as Two-Way Sliding Window (TWSW) data filtering [138] [137] [134]. Data filtering at nodes binds maximum expected and minimum required variation in data report from previously disseminated data to make it eligible for dissemination. Depending upon the type of scenario (and data generated thereof), difference can be calculated simply by comparing aggregated data values or is based on some complex spatial and temporal correlation among different data. The mean of filtering window drifts with time according to observed parameter(s). Since inter-node transmissions are much costlier than processing at a node, hence by performing some in-network processing (i.e. data filtering) at active SNs a large number of inter-node transmissions are avoided by blocking injection of spurious and redundant data towards sink.

Point to be noted here is that SN is term used in general for any type node which can either work as cluster head or as child node for some cluster. SN which acts as child node for a cluster head is termed here onwards as Common Node (CN) to differentiate it from a SN that operates as cluster head and is known as CHN (CHN).

3.2. Two-Way Sliding Window (TWSW) Filtering

3.2.1 Objectives

The overall objective of proposed data suppression using TWSW is to achieve energy savings by reducing number of inter-node transmissions. This is achieved by blocking undesired (redundant and spurious) data as early as possible nearer to data source. To achieve overall objective of network energy efficiency, immediate objectives of our proposed approach are:

- To exploit processing capability of SNs to perform some in-network processing on data generated by onboard sensors before transmitting it further towards sink.
- To define application specific upper and lower limits on variation in consecutive data reports at a node (active SN and/or CHN) to mark it as redundant or spurious data.
- Based on above limits, to use a filtering function to set filtering windows at active SNs and CHNs that block/drop data reports that are redundant or spurious.
- Some times some temporary abnormal conditions exist around few active SNs which do not reflect actual event behaviour. Readings from these active SNs slowly drift away from rest of active SNs and due to comparison with its own preceding readings these nodes are not able to detect this abnormal shift by themselves. Hence, one of the objectives of this approach is to minimise and isolate the effect of this abnormal drift.

Table 3.1 summarizes symbols and notations used in this and next sections.

3.2.2 Assumptions and System Environment

Proposed data filtering approach can be applied to any kind of network topology with different types of SNs and with varying routing protocols underneath. Therefore, to explore the benefits of proposed data filtering approach alone without complicating it with factors like type of topology, routing protocol, node types etc, we take simple network topology with assumptions and system environment are summarized as follows:

- The sensor field is flat, spanning in two dimensional plane.
- All SNs are homogeneous capable of communicating with each other through radio signals using omni-directional antennas. Each node is capable of communicating in dual mode i.e. in low power radio mode and high power radio mode. The

communication between two nodes is symmetric (i.e. if I can hear you, you can here me too).

- Once deployed, SNs do not change their location, however sink and event may move slowly i.e. and thus there is no guarantee that sink/event location at time t and time $t + \tau$ is same.
- Each SN is assumed to be aware of its own location i.e. it knows its geographical coordinates (x, y) which serve as node identifier. Since we assume SNs stationary, assigning coordinates to them is one time job during deployment phase.
- There also exists a sink or access point node (APN) in the network which is a SN nearest to a known geographic location. Inline with assumptions made in many related WSN deployments [9] [139] [102], we take mid point of the area of observation as access point and SN closest to this point as APN or sink (Figure 3.1). This APN acts as a gateway between sensor field and outside world.

Table 3.1. Symbols and Notations Used TWSW

Symbol	**Definition**
CN	Common Node is a SN (SN) other than CHN and sink.
CHN	CHN is a SN that acts as coordinator and leader for a cluster of CNs.
APN	Access Point Node is a term used to represent node to which all CHNs send data. This is synonym for sink in present work.
CNSI	Common Node Sensing Interval is a period after which active CNs sense event
CNDI	Common Node Dissemination Interval defines a period after which CNs disseminate data to CHN.
CHDI	Cluster Head Dissemination Interval defines a period after which CHNs disseminate data to sink and is much larger than CNSI.
T_{ob}	Total period of observation.
β_{CN} / β_{CH}	± Maximum variation allowed for Spurious Data Filtering at CN/CHN.
$\delta_{CN} / \delta_{CH}$	± Minimum required variation for Redundant Data Filtering at CN/CHN.
$CN_{buffer} / CH_{buffer}$	Data buffer at CN/CHN.
CN_{Mean}/CH_{Mean}	Sliding mean of Filtering Window at CN/CHN.
CN_{Avg} / CH_{Avg}	Average of fresh data accumulated at CN_{buffer}/CH_{buffer}.
CN_{UBs} / CH_{UBs}	Upper bound for spurious data filtering at CN/CHN
CN_{LBs} / CH_{LBs}	Lower bound for Spurious Data Filtering at CN/CHN
CN_{UBr} / CH_{UBr}	Upper bound for Redundant Data Filtering at CN/CHN
CN_{LBr} / CH_{LBr}	Lower bound for Redundant Data Filtering at CN/CHN

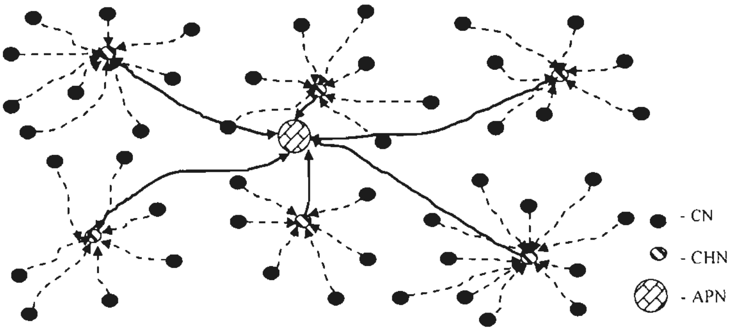


Figure 3.1. Network topology and Clustering

3.2.2.1 Clustering

A simple clustering strategy similar to LEACH [148] is used for initial CHNs selection uniformly across entire sensor field (Figure 3.1). However, it differs from LEACH in CHNs rotation in a way that it does not rotate roles of CHNs among all CNs for energy balance. This is done to keep network topology simple and remain focused on evaluating energy savings achieved using proposed data filtering scheme alone without solving intricacies involved in rotating CHNs for load balancing. Each CHN broadcasts its presence in its low power transmission range using special packet containing its identity (i.e. coordinates) and its remaining energy level. Each CN in the range hears these broadcasts during setup time and forms a set S comprising of coordinates of CHNs from which it hears these broadcasts. Each CN with coordinates (x,y) then calculates minimum distance as follows:

$$d((x,y), S) = \min\{ \sqrt{(x - x_i)^2 + (y - y_i)^2} \ : (x_i, y_i) \in S\} \qquad (3.a)$$

Accordingly, the coordinates in S which gives minimum distance to (x,y) are selected and corresponding CHN is chosen as cluster head for that CN. CN sends its selection result to this CHN and CHN enters CN's coordinates in its child list. If more than one CHNs are at same distance than tie is broken in favour of CHN having maximum residue energy. Further, if there is also tie on residue energy, CHN is selected randomly among these candidate CHNs.

As each node can act in dual radio mode (i.e. low power radio mode and high power radio mode), CNs communicate using low power radio with their CHNs, whereas CHNs communicate with APN using high power radio. As shown in Figure 3.1, APN is assumed to be reachable from every CHN so that CHNs can disseminate data directly to it and form a back bone for data dissemination.

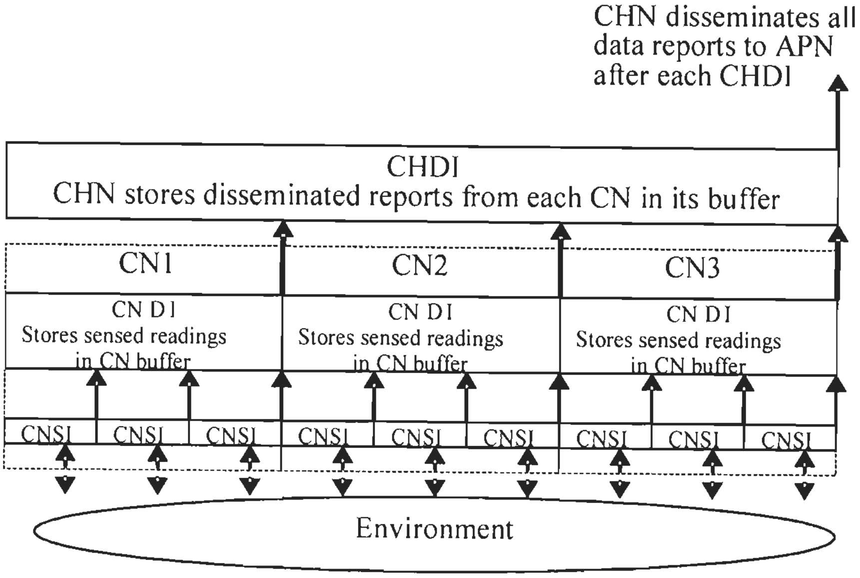


Figure 3.2. Data Dissemination through a single CHN

3.2.2.2 Monitoring and Dissemination Model

The dissemination model is of continuous monitoring type where CNs sense environment continuously over a long period of time. Each CN keeps on sensing the environment after every small period defined by common-node-sensing-interval (CNSI) and stores the values in its small buffer. After expiry of another interval called as common-node-dissemination-interval (CNDI) which is much larger than CNSI, CN prepares itself to disseminate the entire buffer (also known as data report) to its CHN. The entire dissemination model at a CHN is given in Figure 3.2. Before actual dissemination, data report is subjected to filtering check at filtering window at CN and accordingly is either disseminated towards CHN or suppressed. CHN pushes data from its child nodes (i.e. CNs) to its own buffer and disseminate it to APN after every cluster-head-dissemination-interval (CHDI). CHDI is much larger than CNDI. CHN clears its buffer after it disseminates values to APN and prepares itself to store new values. As per application requirement, both CNs and CHNs disseminate each individual reading in buffer without any aggregation like average, min, max etc. Table 3.1 shows symbols used here and in sections next to come.

3.2.3 Proposed Approach

Every application that requires sensing and data dissemination over a very long period of time puts an upper and lower bounds on the granularity of the data required at collection point i.e. sink. This application specific granularity decides the amount of data to be injected in to the network. It extends in two dimensions; one that defines number of

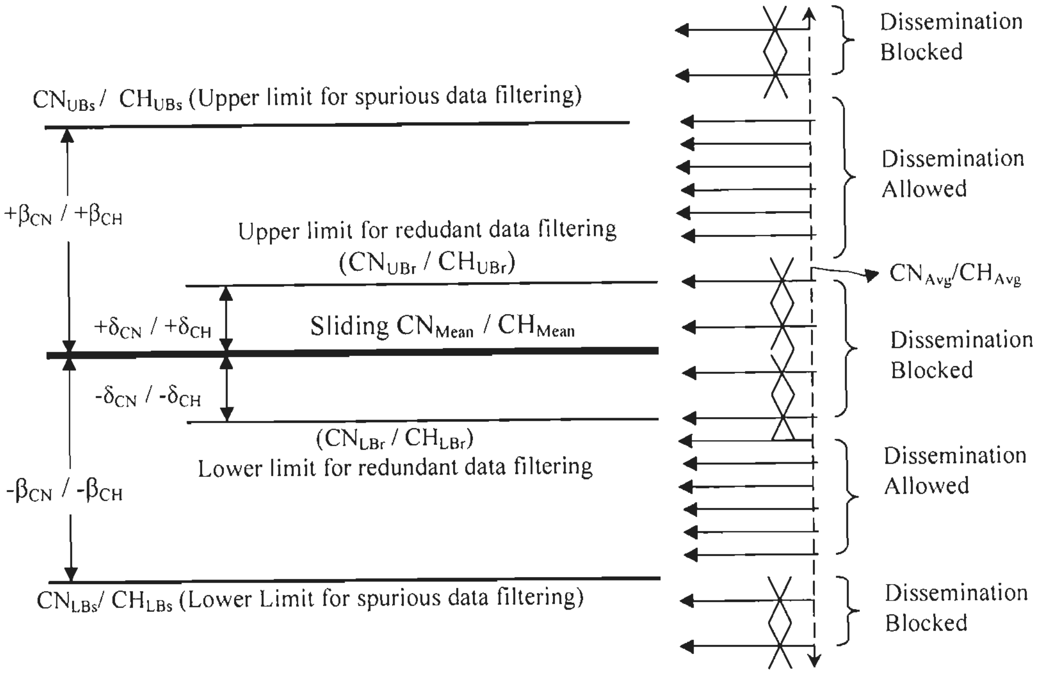


Figure 3.3. Two-Way Sliding Filtering Window at a CN/CHN

samples per unit time and other that defines minimum required and maximum expected variability between current values and previous values. For current study, first dimension is taken care of by defining suitable CNSI and CNDI at CNs and at CHNs. However, later we further explore the possibility of dynamically altering these intervals to gain further energy savings. The second dimension is the focal point of proposed filtering approach where relation between consecutive data reports at a node decides whether to disseminate data further towards sink or block/drop it.

The proposed approach uses a filtering window at a node which binds maximum permissible and minimum required variation in data report from previously disseminated report to make it eligible for dissemination. Filtering is two-way since it suppresses data report which varies either insignificantly or beyond maximum permissible limit from previously disseminated report. Filtering window can be abstracted as in Figure 3.3 with its mean set to average value of all samples in a previously disseminated data report. Also, mean slides in either direction with time and is taken as reference for computing upper and lower limits for filtering window. Since, in present work, we consider lossless dissemination both at CNs and CHNs, entire data reports (i.e. all samples in reports) are disseminated as such and not their aggregated values. However, here average of all samples in a report is calculated to set mean and bounds for filtering window. However, different deployment scenarios will use different correlation function values to set filtering window.

In present work, the purpose of taking average value of data report at a node for comparison is to remain focused on filtering strategy rather than solving intricacies involved in finding complex spatial and/or temporal correlation among different data items with in a report or among data reports from different active SNs.

3.2.3.1 Two-Way Sliding Window Data Filtering at CN

Filtering window filters data in two ways, known as Spurious Data Filtering (SDF) and Redundant Data Filtering (RDF) as defined below:

Spurious Data Filtering (SDF): When dissemination occurs first time, the average of data report at a CN (CN_{Avg}) is set to mean of filtering window at that node (CN_{Mean}). CN_{Mean} then acts as reference point for setting upper and lower bounds for SDF in the filtering window. Bounds for SDF window at CN are set as follows:

Upper bound for SDF at CN: $CN_{UBs} = CN_{Mean} + \beta_{CN}$ and

Lower bound for SDF at CN: $CN_{LBs} = CN_{Mean} - \beta_{CN}$,

where $\pm \beta_{CN}$ is maximum deviation expected by application.

If average of freshly accumulated data report CN_{Avg} is larger than CN_{UBs} or lesser than CN_{LBs}, it indicates spurious data due to malfunctioning of some of nodes or false localized effects near some of nodes. According to application scenario, maximum expected deviation in the CN_{Avg} from CN_{Mean} is $\pm \beta_{CN}$. Therefore, for any new buffered values with CN_{Avg} outside CN_{UBs} or CN_{LBs}, dissemination to CHN is blocked and CN_{buffer} is immediately cleared. For CN_{Avg} between CN_{UBs} and CN_{LBs}, buffered data is disseminated subject to Redundant Data Filtering (explained next). After dissemination CN_{buffer} is cleared immediately, but CN_{Mean} and filtering bound values (i.e. CN_{UBs} and CN_{LBs}) are retained to be used in next data filtering. In this way, SDF avoids dissemination of spurious data and thus reduces number of inter-node transmissions in the network.

Redundant Data Filtering (RDF): When dissemination occurs first time, CN_{Mean} is set to CN_{Avg} and then it acts as a reference point for setting upper and lower bounds for RDF. If new CN_{Avg} of freshly accumulated data report in CN_{buffer} deviates too little from CN_{Mean} (i.e. previous average) and granularity of observed information required by application is such that it does not differentiate between the two, disseminating this data report is unnecessary. If minimum deviation differentiable to application from previously disseminated report is $\pm \delta_{CN}$, then upper and lower bounds for RDF are set as follows:

Upper bound for RDF at CN: $CN_{UBr} = CN_{Mean} + \delta_{CN}$ and

Lower bound for RDF at CN: $CN_{LBr} = CN_{Mean} - \delta_{CN}$,

where $\pm \delta_{CN}$ is the minimum deviation differentiable to application. Any new data report with CN_{Avg} between CN_{UBr} and CN_{LBr} is suppressed and data with CN_{Avg} outside this is disseminated subject to passing SDF.

β_{CN} can be set using general heuristics on historical data patterns available about the application in hand. For example, if network is deployed for measuring soil temperature in a field, there can not be 5°C change in temperature readings with in CNDI of 25 seconds. However, it is possible to have 0.05°C change in temperature after the expiry of same CNDI. Therefore, β_{CN} can be suitably approximated to include valid variation in temperature with in given CNDI. At the other end, δ_{CN} defines bounds for minimum variation required for new data to become eligible for dissemination, therefore is purely defined by application requirements.

Flowchart in Figure 3.4 gives detailed filtering strategy at CN. CN senses phenomenon after every small sensing interval (i.e. CNSI) and is determined by application requirements. If event is detected, sensed data reading is pushed into CN_{buffer}, otherwise nothing is done. Sensed readings are periodically pushed into CN_{buffer} until CNDI expires or CN_{buffer} is full. However, CNSI and CNDI are chosen in such a way that CN_{buffer} never overflows even if event is detected in every sensing attempt. Therefore, overflow check prior to CNDI is unnecessary and hence not considered. CN_{buffer} is then processed to calculate the average (i.e. CN_{Avg}) of data values stored in it. If dissemination is first time, the filtering check is bypassed and CN disseminates buffered values to its CHN and CN_{Mean} of the filtering window is set to CN_{Avg}.

Generally, CNDI is much larger then CNSI and is selected keeping in mind factors like available memory at CN, available memory at CHN, delay requirements of application in hand and CNSI. Despite the fact that CNDI is much larger then CNSI, there are chances that on expiry of dissemination period no values have been pushed in to the CN_{buffer}. This may be due to the fact that CN is far from phenomenon or due to some temporary interference or malfunctioning of node. Hence, on expiry of CNDI an extra check is made to see whether buffer contains some valid data for dissemination or not. In case CN_{buffer} is empty, there is no need for further processing and dissemination is immediately blocked.

Contents of CN_{buffer} are immediately cleared after dissemination to CHN, so that CN readies itself for accumulating fresh data. However, CN_{Mean} and filtering bound values

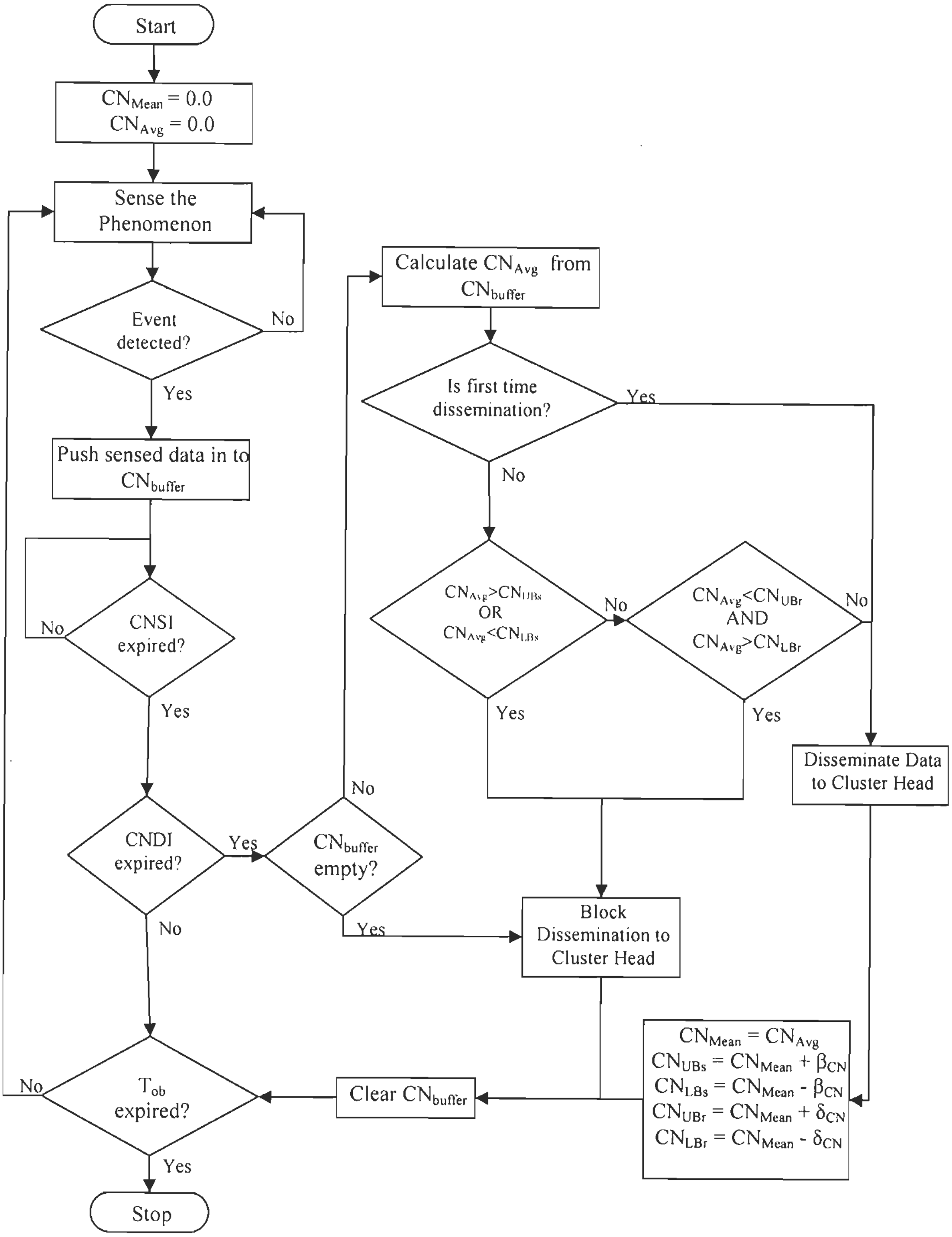


Figure 3.4. Flowchart of Two-Way Sliding Window Data Filtering at CN

(i.e. β_{CN} and δ_{CN}) are retained. In subsequent disseminations CN_{Avg} is calculated from buffered values and undergoes through filtering checks prior to dissemination. If it qualifies filtering criteria, dissemination to CHN is allowed, otherwise dissemination is blocked. This cycle continues till the end of the observation period (i.e. T_{ob}).

To perform both SDF and RDF at CN, the overall filtering criterion becomes:

```
if $((CN_{Avg} > CN_{UBs} \,||\, CN_{Avg} < CN_{LBs}) \,||\, (CN_{Avg} < CN_{UBr} \,\&\&\, CN_{Avg} > CN_{LBr}))$
    {Dissemination Blocked}
else
    {Dissemination Allowed}
```

In case, dissemination is blocked, the recipient treats previously received data as the current data from source.

Zone Effect

There are situations when few active nodes are under the influence of temporary abnormal conditions with respect to actual event behaviour. This is termed as *"zone effect"*. Zone effect may be caused by various reasons like shadowing effect i.e. when channel between event and active SN is partially hindered by some other external agent. Also, few nodes may fall in a region where event does not reveal its true behaviour. For example, if WSN is deployed for accurately measuring field surface temperature during specific period of a day for agricultural research and few SNs fall in a water logged region caused accidentally or fall in a region of shadow caused by some external object. These nodes generate readings which are not in line with rest of active SNs. However, after elapse of each dissemination interval, data readings at these nodes may pass filtering criteria at their local filtering windows and modify its mean accordingly. This is due to that fact that prior, a node compares average of freshly accumulated data in its buffer only with mean of its own filtering window and has no interaction with other nodes (its siblings), hence lacks global knowledge. The rate of change in the value of mean of filtering window at these nodes is however different from rest of nodes in sensor field. Thus, mean of filtering windows in these nodes slowly drifts away from rest of active nodes without being detected locally by filtering process at respective nodes. Therefore, to overcome such zone effect caused by local decisions at individual nodes, data reports must be correlated somewhere higher in network topology to isolate such abnormal reports. Hence, TWSW data filtering is used with some modifications at CHNs which combines data from multiple nodes and provides a broader view for filtering decision.

Apart from dealing with zone effect, data filtering at CHNs further reduces the bulk of data to be transmitted towards sink. At CHN, multiple data reports from many active SNs may be combined into a single larger report and transmission is accomplished in one go instead of multiple transmissions for individual data reports from active SNs.

Also, redundant and spurious data reports can be filtered/dropped similar to filtering at CN level to gain further energy savings.

3.2.3.2 Two-Way Sliding Window Data Filtering at CHN

Figure 3.5 gives detailed algorithm for data filtering at CHN. Unlike CN, which gathers data periodically from environment after every CNSI, CHN keeps on listening to disseminations from its child nodes (i.e. CNs). On hearing dissemination from a child CN, CHN pushes received data into its own buffer i.e. CH_{buffer}. Generally, CH_{buffer} is much larger than the CN_{buffer}, so that it can hold values disseminated from each of its child CN. However, the actual numbers of CNs in its cluster are variable and generally unknown. Therefore, unlike CN_{buffer}, there is probability that CH_{buffer} may overflow due to disseminations from more CNs than anticipated. CHN prepares for dissemination when CHDI expires or CH_{buffer} is full. Similar to CN, it also passes through a check to see whether CH_{buffer} is nonempty or not. In case CH_{buffer} is empty, dissemination is immediately blocked, else it calculates average (i.e. CH_{Avg}) of values stores in it. If CHN is disseminating first time, it is allowed to disseminate to AP without going through filtering check and CH_{Mean} is set to CH_{Avg}. In subsequent disseminations CH_{Avg} is calculated from CH_{buffer} and undergoes through filtering checks prior to dissemination. If it qualifies filtering criteria, dissemination to AP is allowed, otherwise dissemination is blocked. This cycle continues till the end of the observation period (i.e. T_{ob}).

Filtering window at CHN is set similar to that of filtering window at CN. The difference lies in the values of upper and lower bounds for RDF and SDF. Filtering bounds at CHN are calculated as follows:

$CH_{UBs}=CH_{Mean}+\beta_{CH}$ *(Upper bound for spurious filtering at CHN)*

$CH_{LBs}=CH_{Mean}-\beta_{CH}$ *(Lower bound for spurious filtering at CHN)*

$CH_{UBr}=CH_{Mean}+\delta_{CH}$ *(Upper bound for redundant filtering at CHN)*

$CH_{LBr}=CH_{Mean}-\delta_{CH}$ *(Lower bound for redundant filtering at CHN)*

β_{CH} and δ_{CH} are also set as per β_{CN} and δ_{CN} above.

Before disseminating to sink, CHN compares average (CN^{i}_{Avg}) of each i^{th} data report in its buffer (which is from i^{th} CN in its cluster) with CH_{Mean}. If it varies beyond $\pm\beta_{CH}$ from CH_{Mean}, CN_i is detected under zone effect. Had it not been under zone effect, it would have blocked data report at its end. CHN isolates all such CNs under zone effect and proceeds further to calculate CH_{Avg} by discarding reports from these CNs. Also, while

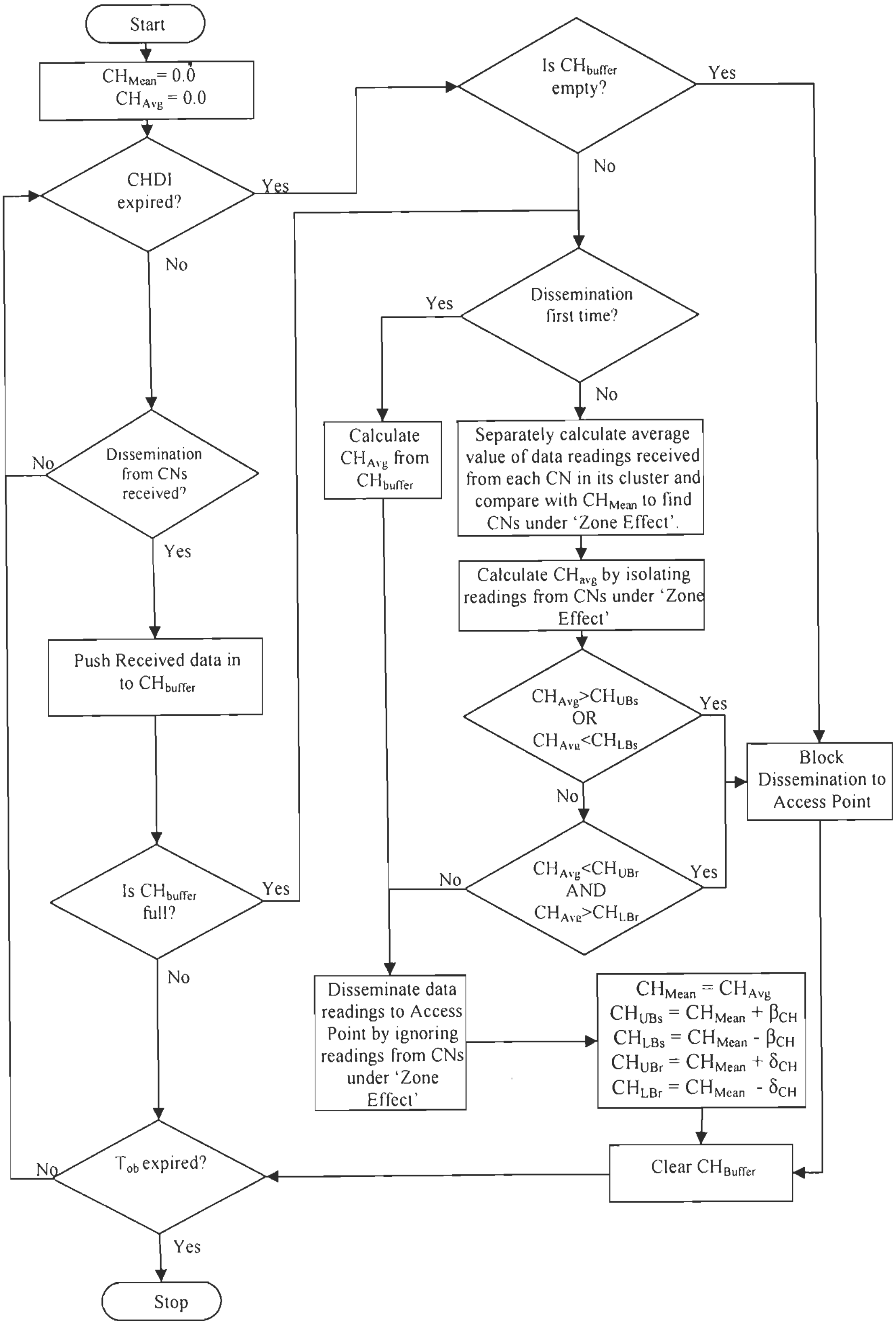


Figure 3.5. Flowchart of Two-Way Sliding Window Data Filtering at CHN

disseminating data to sink it ignores reports from CNs under zone effect and thus avoids forwarding of spurious data and at the same times conserves network energy.

Rest of the algorithm is same as filtering at CN with only difference in the value of CHDI and values of upper and lower bounds for SDF and RDF (i.e. β_{CH} and δ_{CH}). To perform both RDF and SDF at CH, an overall filtering criterion becomes:

```
if(($CH_{Avg} > CH_{UBs}$ || $CH_{Avg} < CH_{LBs}$) || ($CH_{Avg} < CH_{UBr}$ && $CH_{Avg} > CH_{LBr}$))
        {Dissemination Blocked}
else
        {Dissemination Allowed}
```

Alternatively,

```
if(($CH_{Avg} <= CH_{UBs}$ && $CH_{Avg} >= CH_{LBs}$) && ($CH_{Avg} >= CH_{UBr}$ || $CH_{Avg} <= CH_{LBr}$ ))
        {Dissemination Allowed}
else
        {Dissemination Blocked}
```

In case dissemination is blocked, the recipient treats previously received values as the current values from the source.

3.2.4 Mathematical Analysis

In this analysis, we focus on energy consumption in disseminating data from source node buffers to APN (i..e. sink). Nodes once switched to active mode remain in same mode for entire observation period. Therefore, energy consumed at interfaces (transceivers) during start up and shutdown is negligible compared to energy consumed in data dissemination, hence ignored. First, we take CHDI as a period for calculating energy consumption by a single cluster and then take summations over entire network for total period of observation T_{ob}. AP is supposed to have unlimited energy and hence not considered in calculation. Each CN senses event after every CNSI during entire CNDI and pushes sensed readings in to CN_{buffer}. Number of times CN senses the phenomenon is given by:

$$t = \frac{CNDI}{CNSI}, \tag{3.1}$$

Let $E_{j,p}$ be the energy consumed in p^{th} sensing of j^{th} dissemination round. Irrespective of the fact whether event is detected or not, sensors in CN sense environment periodically after each CNSI. So, energy consumed by CN during single (j^{th}) CNDI is:

$$E_j = \sum_{p=1}^{t} E_{j,p} \tag{3.2}$$

Also, each CN disseminates k number of times during one CHDI where k is given by:

$$k = \frac{CHDI}{CNDI}, \tag{3.3}$$

Thus, energy consumed by CN in sensing event during one CHDI is:

$$E_{Sens} = \sum_{j=1}^{k} E_j = \sum_{j=1}^{k} \sum_{p=1}^{l} E_{j,p}, \tag{3.4}$$

In practice, sensors may or may not detect the event due to interference or event not being in sensor range. If event is not detected, no data reading is pushed into CN_{buffer} and thus number of data readings in CN_{buffer} is variable. Let $d_{i,j}$ denotes number of data readings in CN_{buffer} of node CN_i just before j^{th} dissemination (after j^{th} dissemination CN clears CN_{buffer}) and let s be the size of single item. If T_x is the transmission power used by transmitter at each CN_i while operating in low power radio mode and T_r is the data transfer rate, then energy consumed by CN_i in one dissemination to its CHN is given by:

$$e_i = \frac{d_{i,j} \,.\, s \,.\, T_x}{T_r}, \tag{3.5}$$

Also, energy consumed by CN_i in disseminating data in CN_{buffer} to its cluster head k times during one complete CHDI is:

$$E_i = \sum_{j=1}^{k} e_i = \sum_{j=1}^{k} \frac{d_{i,j} \,.\, s \,.\, T_x}{T_r}, \tag{3.6}$$

Suppose, N be the total SNs in the network and out of which N_{CH} nodes are chosen as CHNs. Although, selection of a CHN is random, they are scattered almost uniformly in sensor field. Hence, approximate number of CNs per CHN is given by:

$$m \approx \frac{N - N_{CH}}{N_{CH}}, \tag{3.7}$$

Energy consumed by all CNs under a single CHN in disseminating data during one complete CHDI is:

$$E_{Diss\_CNs\_to\_CHN} = \sum_{i=1}^{m} E_i$$

$$E_{Diss\_CNs\_to\_CHN} = \sum_{i=1}^{m} \sum_{j=1}^{k} \frac{d_{i,j} \,.\, s \,.\, T_x}{T_r}, \tag{3.8}$$

Accordingly, if R_x is the receiving power at CHN, then energy consumed by a single CHN to receive all disseminations from its child CNs during single CHDI is:

$$E_{Rec\_CHN\_from\_CNs} = \sum_{i=1}^{m} \sum_{j=1}^{k} \frac{d_{i,j} \,.\, s \,.\, R_x}{T_r}, \tag{3.9}$$

CHNs communicate with APN using high power radio. Let T'_x (much higher than T_x) be the transmission power of high power radio at CHN and T'_r is its data transfer rate. It accumulates data from its cluster CNs during entire CHDI and then disseminates in one transmission to APN. Let $D_{i,j}$ be the number of disseminations received by CHN from CNs in its cluster and S be the size of each dissemination received. Hence, energy consumed by a CHN to disseminate data received from its child CNs to APN after single CHDI is:

$$E_{Diss\_CHN\_to\_AP} = \frac{D_{i,j} \, . \, S \, . \, T'_x}{T'_r} \qquad (3.10)$$

Now total energy consumed by a cluster during a single CHDI in all network operations can be calculated. Network operations are capturing event observations through periodic sensing, disseminating buffered data from CNs to CHN, receiving disseminations at CHN and on expiry of CHDI disseminating data received from CNs in its cluster to APN or sink. Therefore, total energy consumed by single cluster (i.e. by all CNs in a cluster and CHN) during one CHDI is given as:

$$E_{cluster} = E_{Sens} + E_{Diss\_CNs\_to\_CHN} + E_{Rec\_CHN\_from\_CNs} + E_{Diss\_CHN\_to\_AP}$$

which becomes:

$$E_{cluster} = \sum_{j=1}^{k} \sum_{p=1}^{t} E_{j,p} + \sum_{i=1}^{m} \sum_{j=1}^{k} \frac{d_{i,j} \, . \, s \, . \, T_x}{T_r} + \sum_{i=1}^{m} \sum_{j=1}^{k} \frac{d_{i,j} \, . \, s \, . \, R_x}{T_r} + \frac{D_{i,j} \, . \, S \, . \, T'_x}{T'_r}$$

$$E_{cluster} = \sum_{j=1}^{k} \sum_{p=1}^{t} E_{j,p} + \sum_{i=1}^{m} \sum_{j=1}^{k} \{d_{i,j} \, . \, s. \, [\frac{T_x + R_x}{T_r}]\} + \frac{D_{i,j} \, . \, S \, . \, T'_x}{T'_r}, \qquad (3.11)$$

As the network is divided into N_{CH} clusters, energy consumed by whole network (i.e. by all clusters) in a single CHDI can be given as:

$$= \sum_{1}^{N_{CH}} \left(\sum_{j=1}^{k} \sum_{p=1}^{t} E_{j,p} + \sum_{i=1}^{m} \sum_{j=1}^{k} \{d_{i,j} \, . \, s \, . \, [\frac{T_x + R_x}{T_r}]\} + \frac{D_{i,j} \, . \, S \, . \, T'_x}{T'_r} \right), \qquad (3.12)$$

During entire period of network operation T_{ob}, each CHN disseminated to APN n number of times which is given as:

$$n = \frac{T_{ob}}{CHDI}, \qquad (3.13)$$

Therefore, total energy consumed by whole network comprising N_{CH} clusters during complete period of observation without any filtering applied is:

$$Ewof = n \, . \sum_{1}^{N_{CH}} \left(\sum_{j=1}^{k} \sum_{p=1}^{t} E_{j,p} + \sum_{i=1}^{m} \sum_{j=1}^{k} \{d_{i,j} \, . \, s \, . \, [\frac{T_x + R_x}{T_r}]\} + \frac{D_{i,j} \, . \, S \, . \, T'_x}{T'_r} \right), \qquad (3.14)$$

This gives energy consumed by all CNs and CHNs in the network. However, energy consumed by all CNs and all CHNs separately may be required for analysis purpose. For a given value of N_{CH} and n computed vide (3.13), energy consumed by all CNs alone during entire network operation (E_{CN}) can be computed from (3.4) and (3.8) as:

$$E_{CN} = n \,.\, \sum_{1}^{N_{CH}} (E_{Sens} + E_{Diss\_CNs\_to\_CHN})$$

$$E_{CN} = n \,.\, \sum_{1}^{N_{CH}} \left(\sum_{j=1}^{k} \sum_{p=1}^{t} E_{j,p} + \sum_{i=1}^{m} \sum_{j=1}^{k} \frac{d_{i,j} \,.\, s \,.\, Rx}{Tr}\right), \qquad (3.15)$$

Also, using (3.9) and (3.10) energy consumed by all CHNs only during entire network operation (E_{CH}) is:

$$E_{CH} = n \,.\, \sum_{1}^{N_{CH}} (E_{Rec\_CHN\_from\_CNs} + E_{Diss\_CHN\_to\_AP})$$

$$E_{CH} = n \,.\, \sum_{1}^{N_{CH}} \left(\sum_{i=1}^{m} \sum_{j=1}^{k} \frac{d_{i,j} \,.\, s \,.\, Rx}{Tr} + \frac{D_{i,j} \,.\, S \,.\, T'x}{T'r}\right), \qquad (3.16)$$

3.3 Performance Evaluation

In this section we evaluate the performance of proposed TWSW data filtering approach using network simulator ns-2. We first define simulation parameters, scenarios created and performance metric used. We then see the effect of various factors like CN density and CHN density on evaluation metrics to measure the effectiveness of proposed filtering scheme.

3.3.1 Simulation Parameters

We assume that surface temperature of an agricultural land is to be monitored continuously over a period of 10 hours. Hence, we consider a flat and square sized two dimensional sensor field of size 500×500m^2 in which SNs are randomly generated and deployed. All nodes are homogeneous and each one has onboard dual mode radio i.e. high power radio and low power radio. To keep uniformity across all related work, power consumption parameters here and in next chapter are set same as used in Chapter 5 in which a new virtual grid construction method named as Grid Based Data Dissemination (GBDD) is proposed. Reason for setting given values is given in Chapter 5. Accordingly, values of SN's power parameters in high power mode are set as: transmitting power ($T'x$)=0.66W, receiving power ($R'x$)=0.395W and idling power=0.035W. Power parameters for low power radio are set as: transmitting power (Tx)=0.066W, receiving power (Rx)=0.0395W and idling power=0.0035W. Node is considered capable to transmit up to 100m (R_H) while in high power radio mode and upto 25m (R_L) in low power radio mode. SN changes its transmission power accordingly while switching between high power and low power radio mode. Table 3.2 summarizes all simulation parameters.

CNSI at CN is set 20 seconds to sense event periodically after every 20 seconds. SN's storage (CN_{buffer}) is limited, but still few of data readings generated as a result of event sensing can be buffered in it. Even if CN_{buffer} is not full, application requiring data may impose limit on delay in data received from a CN such that data must not unnecessarily be delayed beyond a period in CN_{buffer} i.e. it must be timely disseminated to its CHN. Thus, we set CNDI as 100 seconds such that buffered data must be disseminated to CHN after every 100 seconds to avoid loss of information. Similarly, data disseminated by a CN is received by its respective CHN and is buffered in its local storage. Alike at

Table 3.2. Simulation Parameters used in TWSW Data Filtering

Parameter/ Entity	**Value**
Type of Sensor Field	Flat two dimensional
Size of sensor field	500 x 500m^2
SN mobility	Stationary
High power radio transmission range (R_H)	100*m*
Low power radio transmission range (R_L)	25*m*
Transmitting power of radio in high power mode ($T'x$)	0.66*W*
Receiving power of radio in high power mode ($R'x$)	0.395*W*
Transmitting power of radio in low power mode (Tx)	0.066*W*
Receiving power of radio in low power mode (Rx)	0.0395*W*
Idling power of a radio	0.0035*W*
Size of filtering window at CN for SDF ($\pm\beta_{CN}$)	±3°*C*
Size of filtering window at CN for RDF ($\pm\ \delta_{CN}$)	±0.03°*C*
Size of filtering window at CH for SDF ($\pm\beta_{CH}$)	±3°*C*
Size of filtering window at CH for RDF ($\pm\ \delta_{CH}$)	±0.03°*C*
Common Node Sensing Interval (CNSI)	20 *seconds*
Common Node Dissemination Interval (CNDI)	100 *seconds*
Cluster Head Dissemination Interval (CHDI)	200 *seconds*
MAC protocol	802.11
Simulation period	10 *hrs*

CNs, buffered data at local storage of a CHN (i.e. CN_{buffer}) must be timely disseminated to sink (i.e. APN) to meet application delay constraints and also avoid buffer overflow. Hence, we set CHDI to 200 seconds. All these intervals are actually scaled down to 1/100 while doing simulation and accordingly power parameters are adjusted. We also assume that a CN and CHN take 0.1 sec to transmit accumulated data.

Size of a filtering window is purely determined by granularity of data required by application running at sink and nature of deployment scenario. If application requires highly granular data, the size of RDF window will be very small i.e. it will block fewer subsequent data readings and allow data with even minute variation to cross. In such a case, $\delta_{CN} \rightarrow 0$ such that $CN_{UBr} \rightarrow CN_{Mean}$ and $CN_{LBr} \rightarrow CN_{Mean}$. Conversely, if low data granularity is required, RDF window size must be large. Larger window size of RDF means upper and lower bounds of RDF window ($CN_{Mean} \pm \delta_{CN}$) have higher values which block more readings with even larger variation from its mean. For current study, we set $\delta CN = \pm 0.03^{o}C$ and keep it uniform across all scenarios simulated.

Similarly, SDF window size is determined by factors like nature of deployment i.e. whether few nodes malfunction due to environmental conditions or channel between sensors and event introduces unexpected errors etc. If these conditions exist, there is every possibility of receiving spurious data readings at CNs and/or CHNs. In such scenarios SDF window must be small to filter undesired data. On the other side, if environmental conditions where event is monitored are smooth and channel between sensors and event is also comparatively error free, SDF window bounds can be relaxed. Intuitively there is possibility that most subsequent readings received are actual measurements of event monitored and not introduced by error. In extreme case, every reading passed by RDF is considered valid no matter how much it varies from mean. Here, $\beta_{CN} \rightarrow \infty$ such that $CN_{UBs} \rightarrow \infty$, $CN_{LBs} \rightarrow \infty$. For simulation, value of β_{CN} is set to $\pm 3^{o}C$.

At CHN level, redundancy in data is introduced by similar views of event by more than one CNs in a cluster during a CHDI. Also, both the communication channels between event and active CNs as well as between CNs and CHNs can suffer from various factors like channel errors, obstruction and temporary abnormal conditions etc. However, the degree of errors introduced varies with the type of communication type like radio, optical or Bluetooth etc. But in present experimentation both channels use similar radio communication and hence have almost same probability of introducing errors. Therefore, filtering window size and bounds are kept same at CN and CHN levels.

Temperature data to represent sensed reading at each CN is randomly generated over entire simulation period with uniform distribution from 15°C to 40°C and then back to 15°C in such a manner that each successive value vary unpredictably from previously generated data within maximum 3°C per CNDI (β_{CN}) limit. However, maximum 10% of CNs are allowed to generate spurious data out side of above distribution.

3.3.2 Performance Metrics

Energy consumed

Energy consumed by a set of nodes is the total sum of energy spent by those nodes in performing required network operation (i.e. transmission, reception, idle state and sensing) during a complete simulation period. Energy consumed is computed for different set of nodes like all CNs, all CHNs and also energy consumed by both (i.e. all CNs and all CHNs). This is done to separately observe the effect of different scenarios on each set of nodes.

Percentage of energy saving

Percentage of energy saving achieved is the less percentage energy consumed by a set of nodes in performing required network operation when TWSW data filtering is used as compared to energy consumed by same set of nodes when no data filtering is used during entire simulation period.

Average delay

Delay is the time taken by a sensed reading to reach at sink from the time it has been sensed at CN. CNSI, CNDI and CHDI all delays its forwarding towards sink. *Average delay* is the value calculated by summing up the individual delays incurred by each data reading and dividing it by the total numbers of data readings disseminated during entire simulation period.

3.3.3 Simulation Experiments

Clustering, path setup and data dissemination model assumed in Section 3.2 are used for setting network topology and path setup between active CNs and APN. Different network scenarios are created and energy savings achieved are evaluated for each scenario at CN level, CHN level and at overall network level. At CN level energy consumed by all active CNs in sensor field is summed up. Similarly at CHN level energy consumed by designated CHNs is summed up irrespective of energy consumed by other nodes such as

CNs and sink(s). For overall network energy consumption, energy consumption at both levels is summed up.

Apart from clustering, path setup and dissemination model as above, LEACH is used for clustering and path setup. With each different scenario, simulation experiments are repeated with this set up also. However, results reveal that energy consumption and savings are same in both cases. This is due to the reason that LEACH mainly differs from above setup in rotating the roles of cluster heads for load balancing. But, in our study we have concentrated on overall energy savings which are same in both cases. Therefore, *EC_wof* in coming sections means energy consumed by WSN when LEACH and our proposed clustering scheme is used without filtering and *EC-wf* means energy consumed by both setups when TWSW data filtering is used. Also, energy consumed calculated theoretically (*EC_th*) vide Section 3.2.4 is also plotted along with simulated results.

Following basic scenarios are created:

Scenario1: *Effect of CN density*

A sensor field of size 500x500 meters, where numbers of CHNs are kept fixed and numbers of CNs are varied from 10 to 100 in steps. This scenario is created to observe the effect of CNs density on energy savings at CN level, CHN level and at overall network level. Apart from studying efficiency of filtering strategy it helps in judging optimum number of CNs that must be deployed in a given sized sensor field.

Scenario 2: *Effect of CHN density*

A sensor field of size 500x500 meters where number of CNs are kept fixed at 50 and a numbers of CHNs are varied from 2 to 10 in steps. This scenario is created to observe the effect of CHN density on energy savings at CN level, CHN level and at overall network level. This scenario also helps in judging optimum number of CHNs with in a given sensor field.

Scenario 3: *Effect of randomness in the behaviour of event*

For a fixed number of CNs (50) and CHNs (4), the rate of change in the temperature per CNDI is varied to see its effect on energy saved by TWSW data filtering at overall network level.

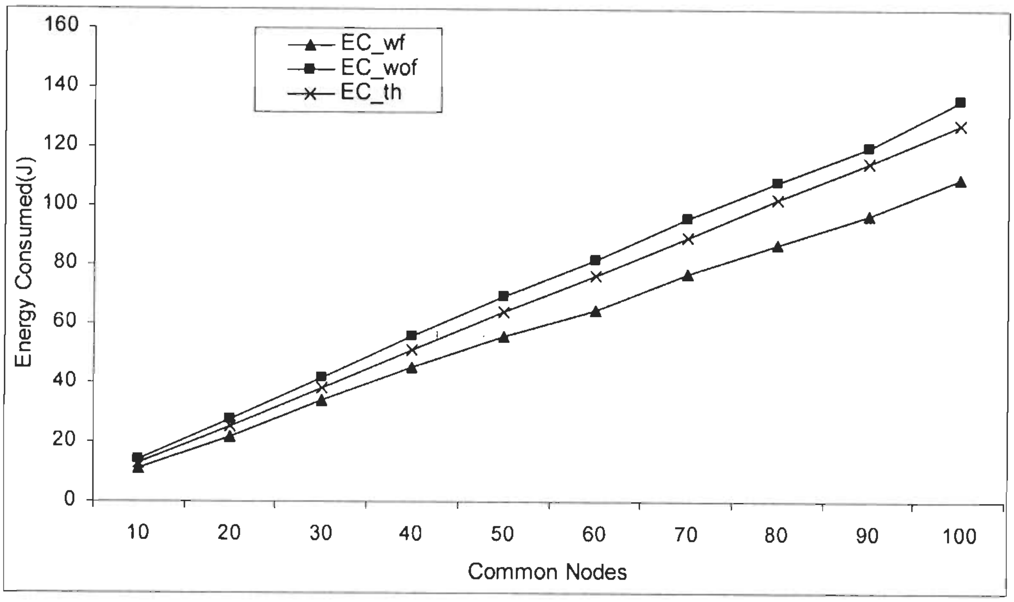


Figure 3.6. Effect of CN density on energy consumed by all CNs when number of CHNs are kept fixed at 4

3.3.4 Effect of CN Density on Energy Consumed at CN Level

To observe the effect of CN density on energy consumed by CNs alone when TWSW data filtering is used, number of CHNs are kept fixed at 4. CNs deployed are varied from 10 to 100 with increments of 10 and at each step we measure separately the energy consumed by all field CNs both when TWSW data filtering is used (*EC_wf*) and when no filtering is used at all (*EC_wof*). Also, energy consumed by all CNs calculated theoretically (*EC_th*) as per Equation (3.15) in Section 3.2.4 is also plotted.Results in Figure 3.6 show that *EC_wof* increases almost linearly with increasing number of CNs. This is due to the simple reason that if number of CNs increases, more CNs sense the event and disseminate data. There is no restriction on number of nodes to disseminate data and any CN having event in its sensing range starts sensing event and disseminating data to its CHN. But when TWSW data filtering is used, due to reduced numbers of disseminations to CHNs from each individual CN *EC_wf* is much less than *EC_wof*. For given simulation settings, average energy savings achieved by TWSW data filtering at all CNs in sensor field range from 21.8% when number of SNs are 10 to 25.2% when number of SNs are 100.

3.3.5 Effect of CN Density on Energy Consumed at CHN Level

Next the effect of varying CN density on the energy consumed by all CHNs present in sensor field is observed. Number of CNs are varied as Section 3.3.4 and energy consumed by all CHNs (fixed number=4) both when TWSW data filtering is used and

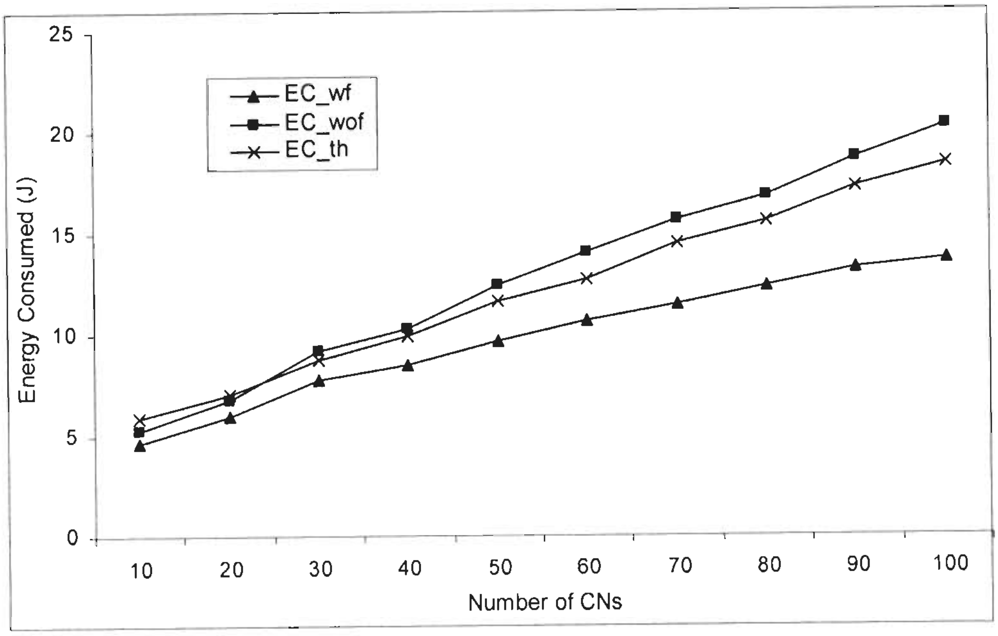


Figure 3.7. Effect of CN density on energy consumed by all CHNs (=4)

when no data filtering is used is measured. Also, energy consumed by all CHNs calculated theoretically (*EC_th*) as per Equation (3.16) in Section 3.2.4 is also plotted. CHNs receive data from CNs in their respective clusters and disseminate it to sink. Point to be noted here is that data received at a CHN has strong relation with the density of the CNs in its cluster. That is the reason why energy consumption studies are carried out separately for CNs and CHNs.

As revealed in Figure 3.7, when number of CNs are less, TWSW data filtering results in less percentage of energy saving. But as number of CNs increases the gap widens and at very high number of CNs, energy saving achieved is quite significant i.e. percentage energy savings at higher CN density is higher than at low CN density. This attributes to the fact that when CN density is low, inter-node distances are large and each CN has its own independent view of event with very less relation with views of neighbouring CNs. This results in lesser data redundancy as defined for RDF in TWSW and hence lesser percentage of data disseminations are blocked resulting in lesser energy savings. As CN density increases, inter-node distances among CNs squeeze. Neighbouring nodes can now have very similar views of the event and hence lot of redundancy in data generated is observed. TWSW data filtering at CHNs thus detects such redundancy in data and blocks unnecessary disseminations to sink. Average percentage energy savings achieved by TWSW data filtering at all CHNs in sensor fields ranges from 11.1% when number of CNs are 10 to 32.66% when number of CNs are 100.

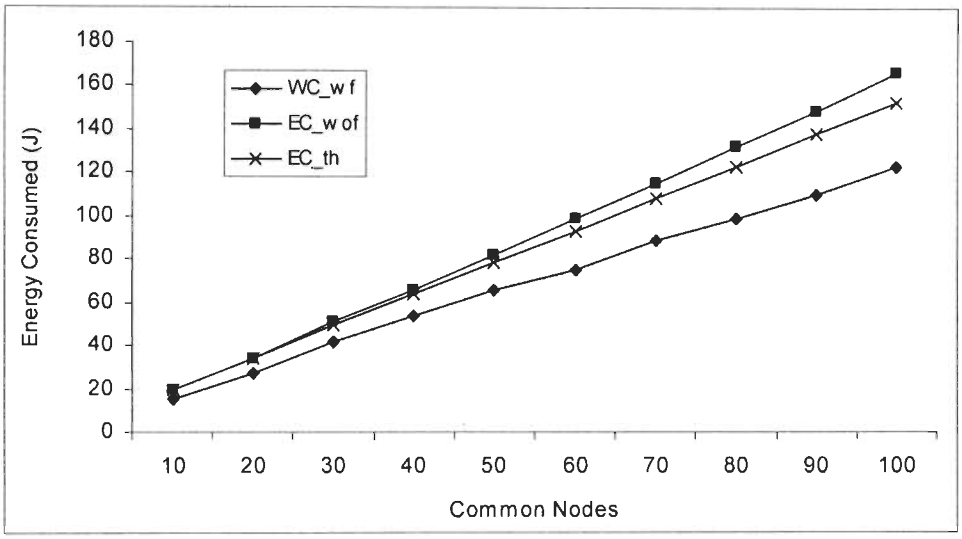


Figure 3.8. Effect of CN density on overall energy consumed by the network

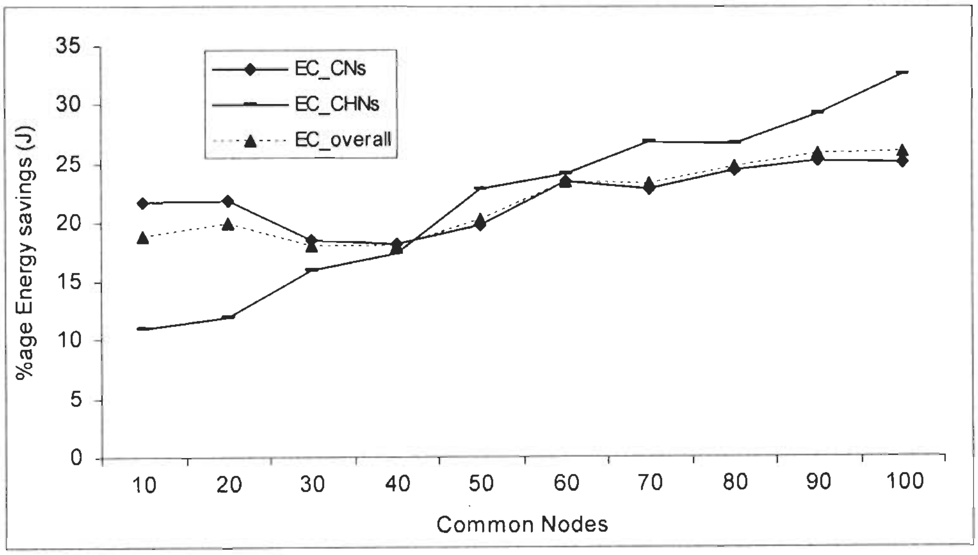


Figure 3.9. Percentage of energy savings at CNs, CHNs and overall network level using TWSW when number of CHNs are fixed (4) and CNs are varied

3.3.6 Effect of CN Density on Overall Network Energy Consumed and Percentage of Energy Saved at CN, CHN and Overall Network Level

Figure 3.8 shows how overall energy consumed by network varies with CN density which closely follows energy consumed at CN level trend due relatively few CHNs as compared to large number of CNs. As shown in Figure 3.9, overall percentage of average energy saved by whole network using TWSW data filtering ranges from 19.8% when number of CNs are 10 to 25.8% when number of CNs are 100. The percentage energy savings at CHNs varies sharply with increasing number of CNs deployment as compared to energy savings at CNs. However, towards overall energy savings, the contribution of energy savings at CNs is major due to their large number as compared to fixed number of CHNs i.e. 4. Therefore, overall average percentage of energy savings follows closely the average percentage energy savings at CNs.

3.3.7 Effect of CHN Density on Energy Consumed at CNs Level

Now we vary number of CHNs and keep number of CNs fixed at 50. This is done to observe the effect of TWSW data filtering on network energy savings achieved when number of clusters are varied with in a defined sensor field having fixed CNs (i.e. number of CNs per cluster will vary). Numbers of clusters formed are equal to number of CHNs deployed. However, the number of clusters and data filtering at CHN level does not have any impact on energy consumed by CNs during entire observation period. Data filtering at CHN only decides whether to block or disseminate data towards sink at its own level and no way dictates to CNs about blocking data disseminations. CNs by themselves run filtering window at their end and sense event periodically (once event comes in its sensing range) independent of their membership to any of CHNs. Therefore, varying number of CHNs does not have any impact on energy savings achieved at CNs using TWSW data filtering. For given present scenario, it remains 55.66 joules irrespective of number of CHNs.

3.3.8 Effect of CHN Density on Energy consumed at CHN Level

As number of CHNs are varied with in a region having fixed number of CNs, the number of CNs per cluster vary. EC_wf and WC_wof are computed through simulation as earlier, whereas EC_th is computed as per Equation (3.16) in Section 3.2.4. It seems apparently that energy consumed at CHN level must increase with their increasing number because of more high power radio transceivers being active. But, as per Figure 3.10, energy consumed by CHNs using TWSW data filtering does not always increase with increasing number of CHNs when TWSW data filtering is used. There is interesting decline in energy consumed initially upto certain number of CHNs and then it increases with increasing number.

Initially, when numbers of CHNs are low, more number of CNs exist per cluster. Also, size of cluster in terms of area covered is larger as compared to scenario when more number of CHNs divide entire sensor region into smaller clusters. This is achieved in most clustering schemes by simple rule that if a CN hears from more than one CHNs it selects CHN with least distance as its cluster head. This implies that in larger sized clusters (i.e in case of small number of CHNs), CNs give view of event from quite wide spread region

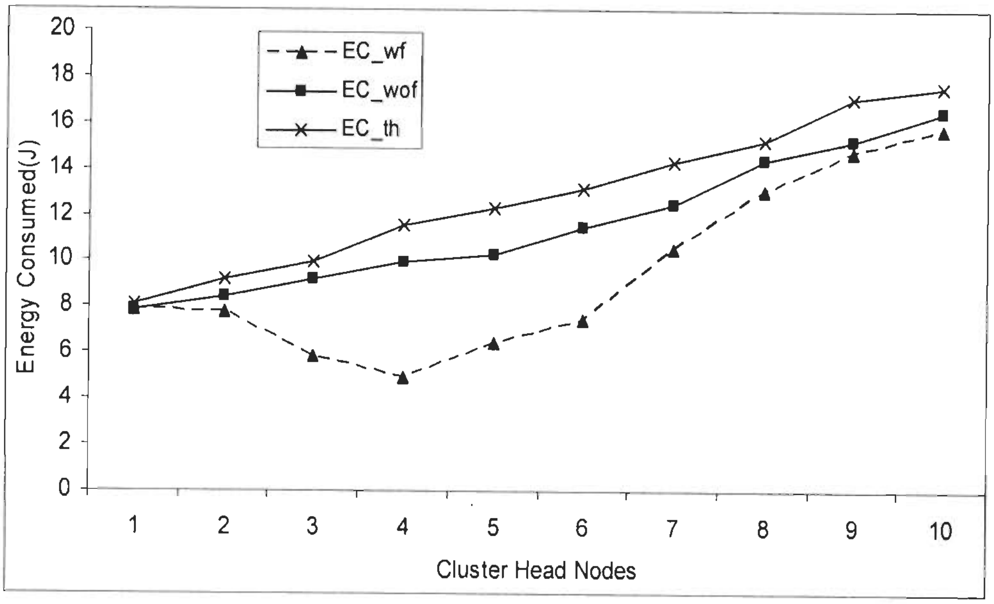


Figure 3.10. Effect of CHN density on energy consumed by network at CHN level

and there are quite high possibility of getting different data readings from CNs as these nodes may have different views of the event. Hence, aggregated average value calculated at a CHN (for filtering check) before dissemination to sink has more chance of passing redundancy check with respect to previously disseminated value (i.e. mean of filtering window). This results in more number of disseminations by CHN per unit time resulting in more energy consumption.

As number of CHNs deployed increases, number of CNs per cluster decreases. Also, CNs in a cluster now belong to smaller region and most of data readings generated are more closely related. Hence, there are more chances of redundant data being blocked (RDF) resulting in more energy savings than savings with larger cluster size. However, with increased CHNs more number of high power radio transceivers become active and consume more energy. But trends in Figure 3.10 clearly show that for small increase in number of CHNs, energy savings achieved due to RDF outweigh extra power consumed due to additional high power active transceivers. Beyond certain number of CHNs, however energy consumption starts increasing due to more number of active high power transceivers and energy savings due to filtering almost vanishes. Hence, from this study one can easily find optimum number of CHNs (i.e. number of clusters thereof) for a given sensor field which in this case is 4.

Apart from energy consumed by high power transceivers at high number of CHNs, there is another reason for this increase in energy consumption. At very large number of

CHNs, number of CNs per cluster decreases. Also, CNs with in a cluster do not have globally synchronized clocks and hence they disseminate data to CHN at different times during a CHN's CHDI. This means that at the time of elapse of CHDI, CHN may have received data only from very few CNs since many of already small number of CNs may have blocked data using RDF or SDF. This accumulated data at CH_{buffer} may be far less than the data that high power transceiver can transmit in one go. Hence, transmission power is wasted.

3.3.9 Effect of CHN Density on Overall Network Energy Consumed and Percentage Energy Saved at CN, CHN and Overall Network Level

Figure 3.11 gives overall network energy consumed with and without TWSW data filtering with varying number of CHNs. *EC_wf* and *WC_wof* are computed through simulation as earlier, whereas *EC_th* is computed as per Equation (3.14) in Section 3.2.4. *EC_wf* here reflects contribution of energy consumed at CN level (which is uniform across different number of CHNs) and at CHN level. Figure 3.12 gives percentage energy saved at CN level, CHN level and at overall network level by using TWSW data filtering at different numbers of CHNs when total number of CNs are fixed at 50. As explained in previous sections and shown in Figure 3.12, when CHNs deployed are varied, the overall percentage energy consumed by network follows trends followed by percentage energy consumed at CHN level. Although, the percentage of saving at overall network level is much less than at CHN level. This is due to the reason that large number of CNs have major contribution towards overall energy consumption as compared to few CHNs and energy consumption at CNs does not have any relation with number of CHNs.

Maximum percentage of energy savings achieved at CHN level by TWSW data filtering is 50% when number of CHNs are 4. At the same number of CHNs, percentage energy saved at overall network level is 23.7% which is also maximum.

3.3.10 Effect of Event Behaviour on Energy Saved

The rate of change of observed event parameter(s) value has huge impact on the energy saved using proposed TWSW data filtering. To perform this study, 50 CNs and 4 CHNs are used as found optimum from above study for the given sensor field. Data readings at CNs are generated in a manner to reflect varying nature of event. For each CN (say CN^i), after elapse of current CNDI (say t^{th}) difference between average value of freshly accumulated data readings in its buffer $CN^i_{buffer,\ t}$ and average of previously accumulated data in its buffer after $t\text{-}1^{st}$ CHNDI is calculated. This difference is averaged

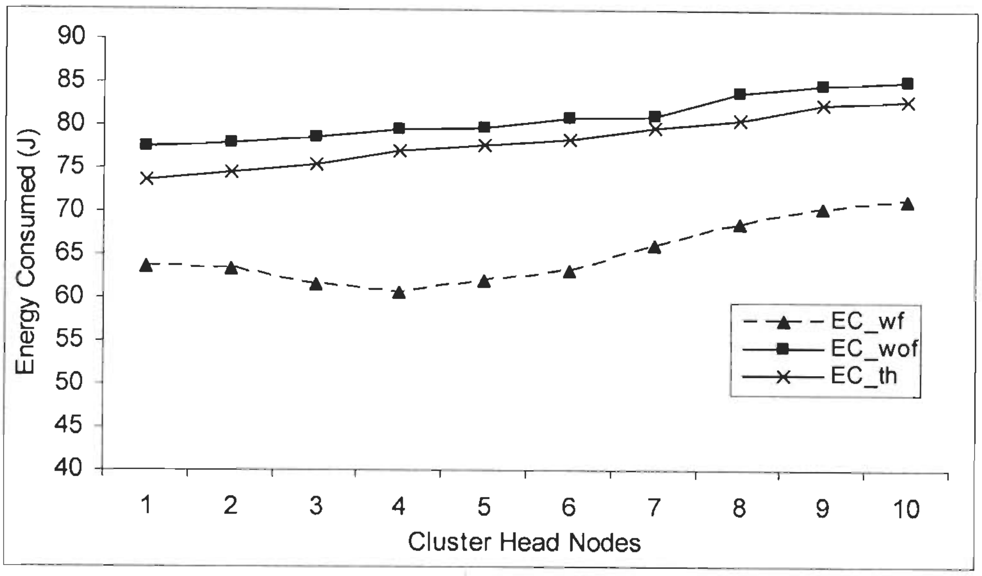


Figure 3.11. Effect of CHN density on overall energy consumed by network

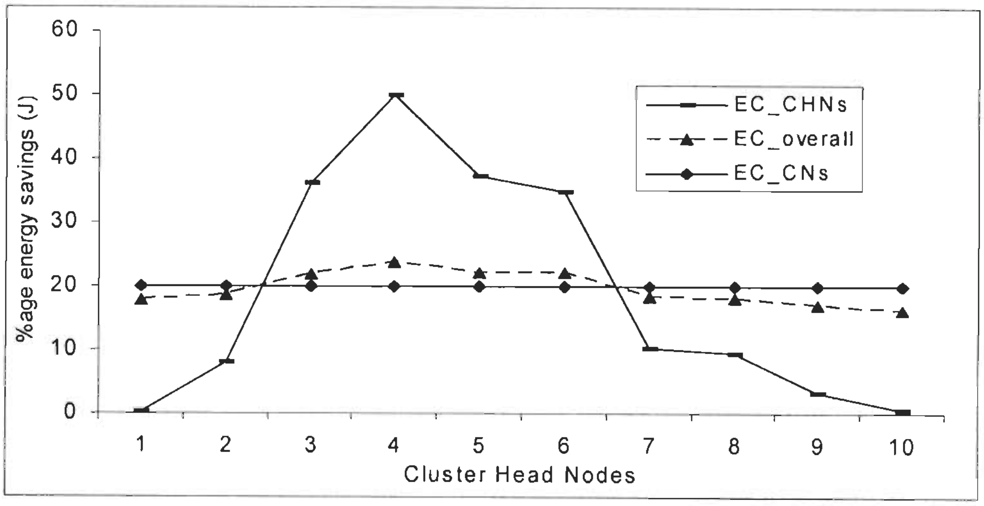


Figure 3.12. Percentage of energy savings at CNs, CHNs and overall network level using TWSW data filtering when number of CNs are fixed (50) and CHNs are varied

across all CNs by summing up the difference and dividing by the total number of CNs. The resulting value is termed as *randomness* and gives the measure of dynamism in event behaviour. Mathematically, randomness in event behaviour during t^{th} CNDI is defined as:

$$randomness = \frac{\sum_{i=1}^{N} (|CN^{i}_{Avg,t} - CN^{i}_{Avg,t-1}|)}{N}, \text{ where } N \text{ is the total number of CNs.}$$

Randomness value is used to put upper limit on data value variation for generating data during entire simulation run. CNs generates data such that difference between successive averages of accumulated data falls uniformly between 0 and value of

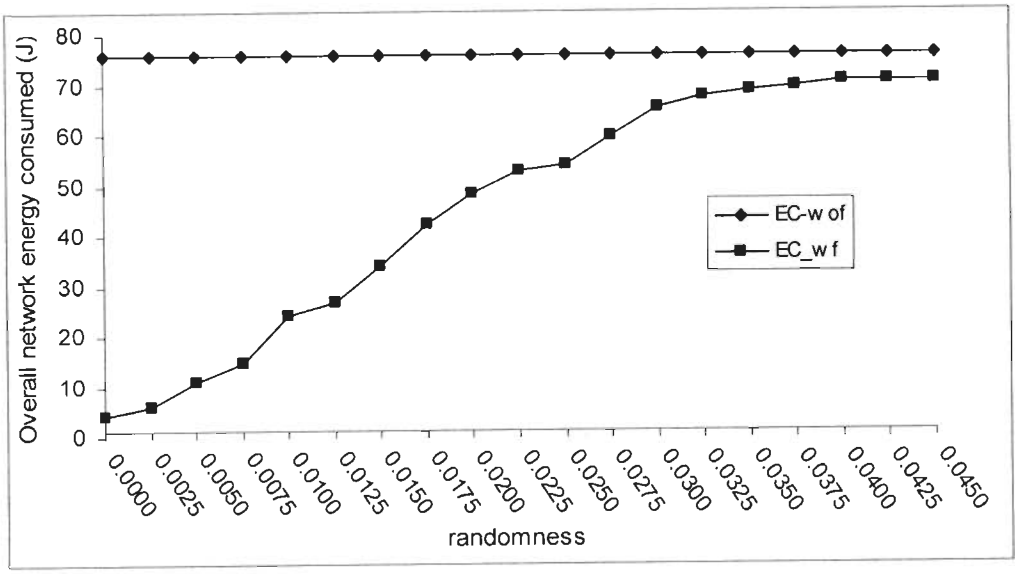


Figure 3.13. Effect of event behaviour on overall network energy consumed

randomness. At first, to mimic event behaviour, we generate data readings such that randomness is almost 0 (i.e. event is static) for all CNs in a sensor field i.e. average value of readings at every CN_{buffer} during every CNDI is almost same as mean of filtering window at that CN. Simulations are run for entire observation period for such event behaviour separately using TWSW data filtering and without any filtering to measure energy savings achieved. Data at CNs is then generated which results in randomness which is slowly varied in steps of 0.005 as 0, 0.05, 0.010, 0.015, ..., 0.03, 0.035 and at each step energy saving due to TWSW data filtering is measured. With increasing randomness value event becomes more dynamic and most of data readings generated passes RDF resulting in more energy consumption.

Figure 3.13 reveals that at very low value of randomness the energy consumed while using TWSW data filtering is very low and at randomness zero it is 94.7% lower than energy consumed without filtering. This is due to the reason that most of successive data readings at each CN are almost same and hence RDF blocks them all avoiding unnecessary transmissions. However, the reason for energy consumed not being 100% less (i.e. 100% energy saved) at randomness 0 is that initially all CNs disseminate first time before mean of filtering window at that CN is set. Also, few spurious data may be generated which passes RDF filtering bounds and also lies in permissible limits of SDF bounds resulting in very few undesired disseminations. At randomness 0.03 and above, overall network energy consumption using TWSW data filtering and without any filtering measures very close. This is due to the reason that ± δ_{CN} (bounds of RDF) is 0.03 and when randomness approaches this value most of data readings at CNs pass RDF filtering resulting in more number of disseminations.

Also, for randomness above 0.0350, the energy consumption using TWSW data filtering becomes constant. Since few CNs may behave maliciously (maximum set to 10%) and generate spurious data which is filtered by SDF resulting in blockage of few disseminations. Hence, energies consumed with and without filtering are not exactly same at randomness 0.0375 and higher.

3.3.11 Effect on Average Delay

Average delay as defined earlier is the average time taken by a data reading generated at a CN (as a result of event sensing) to reach at sink. Interesting point to note here is that if a receiver (sink or CHN) does not receive data dissemination from a sender during currently scheduled interval, it assumes previously received value as the value for currently scheduled dissemination. Sender (CN or CHN) might not have sent new data possibly due to data being filtered. If data is filtered, it means that no interesting data that exist at sender to disseminate i.e. either data is redundant or caused by external agent not related to event (spurious). In either case, previously disseminated data is the representative of current dissemination. Therefore, in current study, logically we suppose that data is transmitted to receiver even if it is blocked at sender because intuitively previous data is conveyed to receiver.

Also, all CNs although have same value of CNSI and CNDI, still they sense and disseminate during different times due to lack of global time synchronization as prevalent in most distributed systems. For, example, a CHN listens to disseminations from CNs in its cluster for 200seconds and receives data reports at different times spread across 200seconds.

In present scenario, where sensing frequency is set one time, all of CNSI (20 seconds), CNDI (100 seconds) and CHDI (200 seconds) remain unchanged during entire observation period. Also, buffer sizes at CNs and CHNs are also fixed. Therefore, any variation in numbers of CNs and CHNs does not have any impact on average delay. Average delay in this case remains around 160seconds. Also, event behaviour (randomness) does not have any impact on average delay since sensing frequency is fixed.

3.4 Conclusions

Most WSN deployments are targeted for continuously monitoring certain environmental conditions or habitats over very long period of time. SNs having limited battery life are deployed randomly in such regions to sense certain event parameter(s). Long spells of continuous sensing without human interaction expect WSN to last long enough to accomplish the monitoring task. In turn, it requires SNs to consume as much less energy as possible to enhance overall network lifetime.

Inter-node transmissions are the major source of energy consumption in WSNs. Therefore, focus of our study in this chapter has been to reduce these inter-node transmissions without missing required information at sink. To reduce these inter-node transmissions, relationship among different data readings gathered at CNs and CHNs is exploited. Instead of injecting and transmitting all data readings into the network without any check, we propose a data filtering scheme called as Two-Way Sliding Window (TWSW) data filtering which exploits relationship among successive data readings to suppress certain redundant and spurious data readings. TWSW filtering window is used both at each active CN level and CHN level.

Redundant data filtering (RDF) avoids disseminating data readings which do not vary at all or vary insignificantly from previously disseminated data readings. These readings do not carry any useful information for application running at sink and hence their filtering/blocking avoids lot of unnecessary inter-node transmissions. For relatively slow changing events, chances of generation of highly redundant data readings are relatively high as compared to more dynamic events.

Also, if some spurious data is generated due to malfunctioning of few CNs or some other temporary abnormal conditions, it is easily identified and blocked using SDF. SDF further adds to network energy savings and maintains quality of data at sink. Results clearly reveal the effectiveness of TWSW data filtering in achieving overall network energy savings. The findings of this study can be summarized as:

1. Prior to dissemination, if certain in-network processing is done on data, a significant amount of it can be reduced. This in turn results in lot of energy savings due to reduced number of inter-node transmissions.
2. TWSW data filtering is most suitable for scenarios where WSN is deployed to monitor slowly changing event(s) spread across a large sensor field for very long period. This is due to the fact that TWSW sets one time the sensing frequency and

dissemination intervals at the start of network operation. If event is more dynamic lower sensing frequency set can miss interesting observations and there is not way to change sensing frequency. Also, slowly changing events over long periods cause more redundant readings which are effectively filtered out by TWSW.

3. For a given number of CNs with in a defined sensor field, there exist an optimum number of CHNs which gives best performance in terms of energy savings. In current study, where sensor field is 500 x 500*meters* with 50 CNs, the optimum number of CHNs found are 4. However, number of CNs deployed are dependent on accuracy and granularity of data required at sink.

Chapter 4

Adaptive Sensing and Dynamic Data Reporting (ASDDR)

4.1 Introduction

Chapter 3 gives a TWSW data filtering approach that achieves energy efficiency by reducing number of inter-node transmissions in a WSN. This is achieved by filtering/blocking undesired data at CNs and CHNs using Two-Way Sliding Window (TWSW) data filtering at these nodes. TWSW can be used with any existing data dissemination approach at CNs and at CHNS strategically selected for a given scenario.

Redundancy in data is one major reason and motive for adopting above data filtering approach. Redundancy can be seen differently at different levels i.e. at CNs and CHNs. At a given CN, if successive data reports generated as a result of event sensing are redundant, then certainly at different times during this period the event has changed its behaviour very little. The difference is either zero or not differentiable to application requiring data i.e. event is very static. Proposed TWSW scheme suppresses all such redundant reports at CN but does not stop/alter CN from sensing the event at successive CNSIs and CNDIs. CN keeps on sensing the event with same frequency after each CNSI till CNDI elapses and then again continues sensing despite the fact that accumulated data report is redundant with respect to previously disseminated report. However, the dissemination to CHN is inhibited. At a CHN, data reports from various CNs from its cluster are received and if two different data reports are very similar in accumulated data readings in it and also their average values differ little, then redundancy is reported. Hence, redundancy at a CHN is caused by two or more data reports generated from different CNs at different times (no global clock)/locations in its cluster. Whereas, redundancy at a CN is caused by similarity between two reports generated at same node.

TWSW data filtering however, deals with both above types of redundancies but it does not alter data reporting intervals (CNDI and CHDI) and no way instructs CNs to vary their sensing frequencies. For example, if event is totally static for some time, there is no use in sensing the event with same frequency which has been once set. If sensing frequency is set very high and later event becomes relatively static, it wastes lot of energy in sensing and disseminating redundant data. Also, if sensing frequency is initially set low and later event behaviour becomes more random and changes rapidly, many useful and interesting readings may be missed and event monitoring may loose accuracy. For example, suppose monitoring/control of temperature variation at a particular area of an industry during operational hours is required and CNSI (sensing interval) is set 10 seconds. When rate of change in the temperature was quite low, the readings generated reveal fair behaviour of an event. Now, if temperature variation becomes random with sudden surges in value, then with same CNSI many of these surges which appear between successive CNSIs are not noticed. This is due to the fact that temperature might have settled to normal value during successive sensing not revealing true nature of event.

Therefore, in this chapter we propose an approach called as Adaptive Sensing and Dynamic Data Reporting (ASDDR) that helps CNs to adapt to changing behaviour of event(s) by adjusting their sensing frequencies [137].[134]. Also, dissemination intervals CNDI and CHDI at CNs and CHNs respectively are altered accordingly. TWSW and ASDDR when used together leads to more energy efficient data dissemination which disseminates only interesting information towards sink (with out compromising granularity of data required) and adapts to changing event behaviour of an event.

4.2 Adaptive Sensing and Dynamic Data Reporting (ASDDR)

4.2.1 Objectives

Like TWSW data filtering, the overall objective here is to achieve network energy savings. But apart from eliminating undesired data, this approach thrives to tune/adjust sensing frequency at CNs on the fly depending upon the behaviour of event and alter reporting frequencies (CNDI alone or both CNDI and CHDI). To meet these broader objectives, immediate objectives of proposed ASDDR approach are:

- To find means to infer the current rate of change in event behaviour.

- To find appropriate point in the network where above information is to be gathered and interpreted.
- To find appropriate and automatic adjustment (without human intervention) value to alter CNSI, CNDI and CHDI.
- To find method to convey above adjustments to appropriate nodes.

4.2.2 Assumptions and System Environment

All assumptions, system environment and network model are exactly same as in previous chapter.

4.2.3 Clustering

Same clustering scheme as used in previous chapter is used.

4.2.4 Data Dissemination Model

TWSW data filtering in previous chapter uses a data dissemination model which sets values of CNSI, CNDI and CHDI once, and do not alter these values during entire observation period. In ASDDR, however the dissemination model is same but value of CNSI is continuously varied to adapt to changing behaviour of an event. Accordingly, CNDI and CHDI are changed as they are integral multiples of CNSI. As will be explained next, for CHDI however there exist two options. In first option, CHDI is kept fixed despite changing CNSI and in other one, CHDI is also varied.

4.2.5 Proposed Approach

Complete setup of TWSW data filtering scheme including filtering windows, data buffers, filtering algorithms at both CN and CHN levels, and data dissemination model is used by proposed scheme. Along with aggregating and checking accumulated data against certain filtering functions, the processing capabilities of SNs (CNs and CHNs) are further exploited to run an additional algorithm that adds dynamism in the scheme to adapt to changing event behaviour. A complete algorithm for dynamically setting sensing and dissemination intervals is given as Algorithm 4.1.

Active CHNs are selected as points in network to analyse and decide the corrective action when event changes its behaviour. CHNs analyses flow of filtered data reports from CNs in its cluster to set sensing/dissemination intervals dynamically. Each active CN has a dissemination interval CNDI which is given as:

$$\text{CNDI} = \text{CNSI} \cdot p, \text{ for some small integer value of } p. \qquad (4.1)$$

Algorithm 4.1. Adaptive Sensing and Dynamic Data Dissemination

```
1.  Input: Initial values for CNSI which is set based on estimated event behaviour
2.  Compute CNDI= j . CNSI, where j is small integer chosen suitably and
            CHDI = m . CNDI, where m is small integer decided based on memory
            available at CHN and/or application requirement
3.  Compute $R_{max}$ = (CHDI/CNDI) . k, where k is the number of active SNs in that cluster
4.  Compute $R_{upper}$ = f . $R_{max}$, where f is a factor representing fairness of channel and probability
    of temporary abnormal conditions near few SNs
5.  Compute $R_{lower}$ = $R_{max}$ / 2
6.  set R = 0                    //Initialize number of data reports
7.  Until (CHDI expires) //Count data reports received during complete CHDI
        i. CHN receives data report sent by a CN from its cluster after TWSW data filtering check
        ii. R = R + 1
8.  Disseminate data in $CH_{buffer}$ to sink after elapse of CHDI
9.  if ( R < $R_{upper}$ && R>= $R_{lower}$)
            Do nothing
            break;
        else if(R >= $R_{upper}$)            //Sensing rate must be increased
            CNSI = CNSI - $\Delta t$, for small $\Delta t$ compared to CNSI
        else                  // Sensing rate must be decreased
            CNSI = CNSI + $\Delta t$, for small $\Delta t$ compared to CNSI
10. Re-compute CNDI = j . CNSI
11. if(variable CHDI is used) //otherwise keep CHDI unchanged
        CHDI = m . CNDI
12. Re-compute $R_{max}$ = (CHDI/CNDI) . k;
                $R_{upper}$ = f . $R_{max}$
                $R_{lower}$ = $R_{max}$ / 2
13. if (values of CNSI and CNDI are modified)
        Broadcast new CNSI and CNDI values with in the cluster
    // Active CNs now sense event after each modified value of CNSI and disseminate to CHN
    after modified CNDI.
14. Reset R i.e. R = 0
15. Until (Observation period has not expired) GOTO step 7
        //Get ready to accumulate data reports during next CHDI
```

Each CHN has a CHDI which is relatively much larger than CNSI and is set as:

$$\text{CHDI} = \text{CNDI} \,.\, m, \quad \text{for some small integer } m. \tag{4.2}$$

If each accumulated report at a CN passes filtering check, then CN disseminates after each CNDI and its CHN receives CHDI/CNDI number of reports from this CN. Hence, if a CHN has k CNs, then maximum numbers of data reports (R_{max}) it receives during single CHDI are:

$$R_{\text{max}} = (\text{CHDI} / \text{CNDI}) \,.\, k \tag{4.3}$$

Active CNs with in a cluster may send different number of data reports to CHN as each has its own view of event from different locations. Some CNs may block its subsequent data reports and others not during approximately same time (each CN has its own clock without being synchronised with others and hence may start CNSI at different time). According to proposed data filtering approach, if event is completely static, all CNs block subsequent redundant data reports. On the other extreme, if event is rapidly changing, CNs generate interesting data during each CNDI and do not block any subsequent data reports. This fact is exploited to set sensing interval so that generation of redundant data reports is avoided at first place and also important points of observation are not missed.

If a CHN during a single CHDI receives R_{max} data reports, it shows highly changing nature of event. This gives intuition that if CNSI is further reduced (i.e. sensing frequency increased), more interesting data reports may be gathered revealing more granular information about the event. Hence, CNSI is reduced by a small amount and revised values of CNDI and CHDI are calculated. CHN conveys new value of CNSI to all CNs in cluster.

Ideally, R_{max} is a number which shows data reports received in a perfect scenario without any irregularity like channel noise and temporary abnormal conditions etc. But in practical scenarios, due to any of these factors subsequent data reports from a CN may be distorted to become redundant or spurious despite the fact that event is highly changing. Therefore, for practical purpose the limit on maximum number of data reports received at CHN is set at some value lower than R_{max}. For a given CNSI, let this limit be called R_{upper} and is calculated as:

$$R_{\text{upper}} = f \,.\, R_{\text{max}} = f \,.\, (\text{CHDI} / \text{CNDI}) \,.\, k \tag{4.4}$$

where *f* is a factor depicting correctness of channels (both from SN to CHN and from event to CN) and probability of temporary abnormal conditions near few SNs. We take value of *f* as 0.9.

Also, if data reports *R* received by CHN during a CHDI are very less compared to R_{upper}, it indicates that many of reports are blocked at CNs. Blocking may be caused by redundancy or injection of spurious data. One can easily infer here that CNs are wasting energy in unnecessary sensing whereas data reports so generated results in redundant reports. Therefore, obvious choice is to increase the value of sensing interval CNSI i.e. to reduce sensing frequency. Like R_{upper}, lower limit R_{lower} on *R* to initiate the process of increasing value of CNSI can be set to some appropriate value and here it is taken as:

$$R_{lower} = R_{max} / 2 = (f \ . (CHDI / CNDI) \ . k) / 2 \qquad (4.5)$$

On expiry of current CHDI and just prior to data dissemination to sink, each CHN counts the number of data reports *R* received from CNs in its cluster and compares it with R_{upper} and R_{lower}. If $R >= R_{upper}$ or $R < R_{upper}$ process to set new CNSI is initiated. CHN calculates new CNSI and broadcasts it to its cluster CNs. If $R < R_{upper}$ and $R >= R_{lower}$, no changes are made i.e. CNs sense event with existing value of CNSI.

If $R >= R_{upper}$, the new value of CNSI is set to:

$$CNSI = CNSI - \Delta t, \text{ where } \Delta t \text{ is small time period compared to CNSI} \qquad (4.6)$$

Value of Δt is decided by appropriately considering the nature of sensing scenario and application requirements. Also, revised value of CNDI and CHDI are calculated.

If during a CHDI, numbers of data reports *R* received by CHN from SNs in its cluster are less than R_{lower} (i.e. $R < R_{lower}$), the value of CNSI is set as:

$$CNSI = CNSI + \Delta t \qquad (4.7)$$

where Δt is same as in Equation (4.4) above. Further, revised values of CNDI and CHDI are calculated.

Reporting Data from CHN to sink

While adaptive sensing is used and accordingly CNSI and CNDI are set, there exist two options in reporting data to sink from CHNs (i.e. setting CHDI).

Varying CHDI: If application running at sink is only concerned in useful data about event and does not enforce any strict reporting frequency, then CHDI can vary according to the data availability from source and/or current value of CNSI (value of CNDI thereof). In present scheme CHDI is varied only in accordance with value of CNSI. This is because

CHDI is integral multiple of CNDI which itself is integral multiple of CNSI. Therefore, change in CNSI due to ASDDR scheme alters both CNDI and CHDI which as per Equations (4.3), (4.4) and (4.5) above keeps R_{max}, R_{upper} and R_{lower} unchanged for changing CNSI. For illustration of this effect and to show effect of dynamic setting of CNSI, an initial small snapshot of simulation is shown in Figure 4.1(a).

Point to be noted here is that to simplify the presentation, along Y-axis two value series of different types are plotted together namely number of reports R received at a CHN and value of CNSI which is time. Also, X-axis represents simulation clock where each point of observation like 6250, 6650 etc represents clock time in seconds at the elapse of a particular CHDI. If CHDI elapsed at 6250 seconds and new CHDI value set is 400 seconds, then next point of observation on X-axis is 6650 seconds. These points of observations after elapse of certain CHDIs are termed here as *Running CHDI's*.

It is clear from the plot that while R crosses either R_{upper} or R_{lower}, CNSI is decreased and increased respectively to adapt to changing nature of event. But when R is between these two limits, CNSI is kept intact as during running CHDI 4650 to 7050. Intuitively, R is kept between these two limits for energy efficient dissemination.

Fixed CHDI: If application at sink needs data periodically from each CHN after a fixed interval irrespective of the nature of event, then CHDI must be fixed. Hence, as per Equation (4.2), CHDI once decided remains intact for entire observation period. Any changes made to CNSI and CNDI does not affect the dissemination interval at CHN. Here care must be taken to decide CHDI in such a manner that if CNSI becomes too small and CNs disseminates too many reports to CHN in a given CHDI, it still is able to buffer those reports. Further, as per Equations (4.3), (4.4) and (4.5) in Section 4.2.5, fixed CHDI with changing CNSI and CNDI results in changing value of R_{max}, R_{upper} and R_{lower}. This scenario is depicted in Figure 4.1(b). Like variable CHDI, our approach here also tries to drag R between R_{upper} and R_{lower}.

Overall the approach tries to keep sensing frequency in a range that helps to capture fine grained details about an event and avoids unnecessary high sensing in a relatively static event scenario.

4.3 Performance Evaluation

In this section we evaluate the performance of proposed ASDDR approach through simulation study. We first define simulation parameters, scenarios created and performance metrics used. We then see the effect of factors like sensing frequency (value

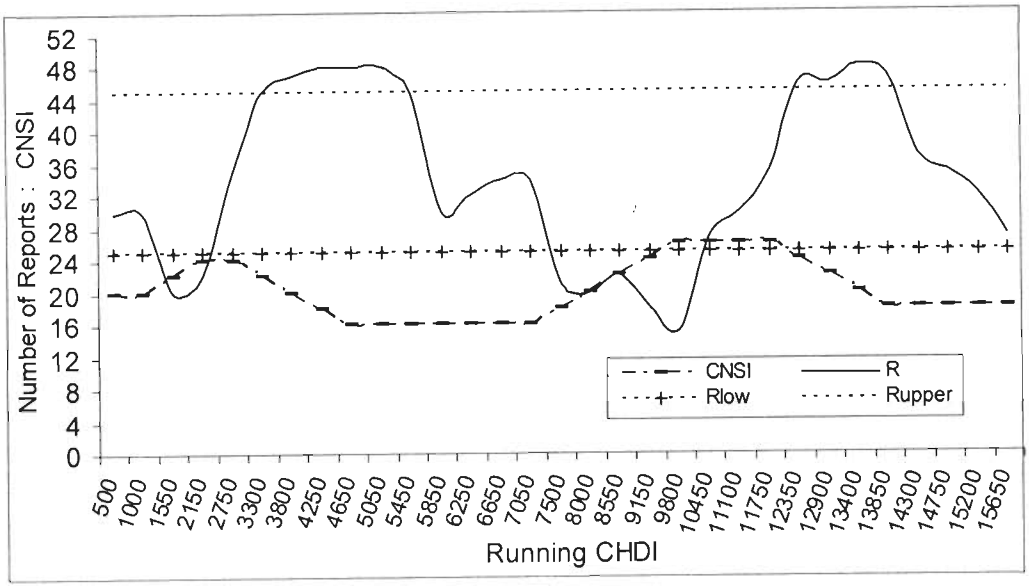


Figure 4.1(a). Altering CNSI with variable CHDI which keeps R_{max}, R_{upper} and R_{lower} fixed

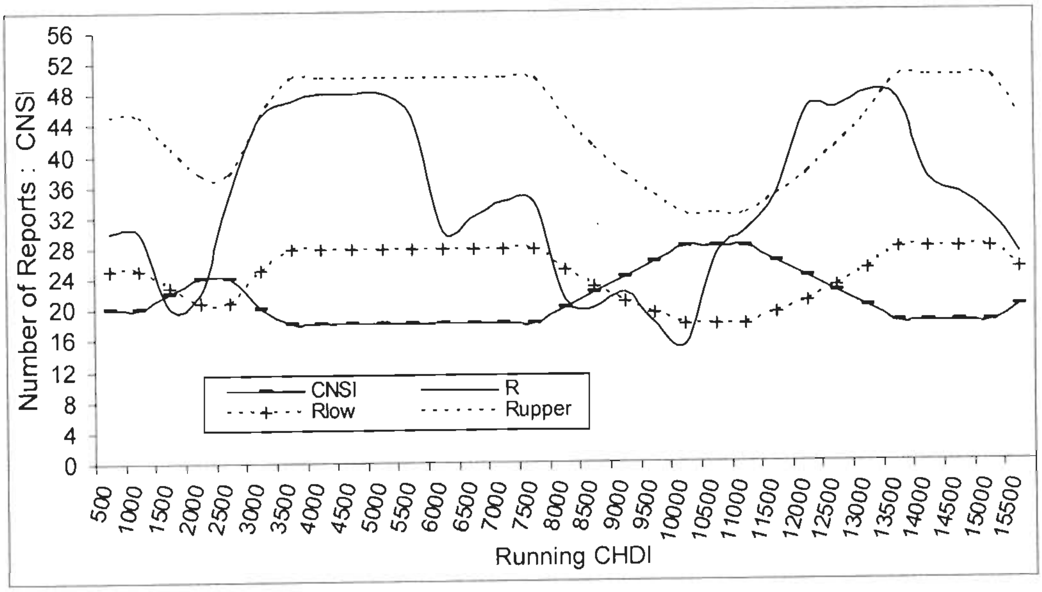


Figure 4.1(b). Altering CNSI with fixed CHDI which makes R_{max}, R_{upper} and R_{lower} variable

of CNSI) and event behaviour (*randomness*) on evaluation metrics to evaluate the effectiveness of proposed filtering scheme.

4.3.1 Simulation Parameters

Simulation parameters used here are exactly same as used in previous chapter except an additional parameter Δt. Δt is a small increment or decrement in the value of CNSI required when rate of change in event behaviour reaches a particular limit. In current study, we take the value of Δt as 2 seconds.

4.3.2 Performance Metrics

Same performance metrics (*i.e. energy consumed, percentage energy saving* and *average delay*) as used in previous chapter are used here. *Energy consumed is* computed for different sets of nodes i.e. for all CHNs, for all CNs and for entire network. However, *percentage energy saving* achieved in present context is the less percentage energy consumed by a set of nodes (i.e by all CNs, all CHNs or both depends on the set selected for measurement) in performing required network operation when ASDDR is used as compared to energy consumed by same set of nodes when no data filtering is used during entire simulation period.

4.3.3 Simulation Scenarios

Scenario 1

To observe the effect of different sensing frequencies (CNSI's thereof) set statically as well as effect of adaptively (dynamically) setting sensing frequency at CNs on network energy consumption, 50 CNs and 4 CHNs are deployed for a sensor field of 500 x 500 meters as found optimum in previous chapter. However, for more clarity same set up is repeated with varying CN density from 10 to 100 with 4 CHNs.

Scenario 2

To observe the effect of *randomness* in event behaviour (as defined in previous chapter) on average delay and energy consumption, 50 CNs and 4 CHNs are deployed. *Randomness* value is varied in steps and performance metrics are measured at each step with dynamic sensing approach both for fixed CHDI and variable CHDI approach separately.

4.3.4 Effect of Different Sensing Scenarios on Overall Network Energy Consumption/Savings

First of all, the effect of different sensing frequencies (set statically) on overall network energy consumption is studied. Energy consumption with different sensing frequencies is measured separately once with TWSW data filtering and once without any data filtering. As per scenario 1, 50 CNs and 4 CHNs are deployed and are kept fixed. CNSI is varied in steps as 5s, 10s, 15s and 20s, while at each step CNDI and CHDI are set accordingly as per Equations (4.1) and (4.2) in Section 4.2.5. Also, at each step simulation is run once with TWSW data filtering and then repeated without any data filtering. This gives the measure of percentage of network energy saved by using TWSW data filtering compared to network energy consumed when same setup is run with no data filtering.

Percentage energy saved is less percentage of energy consumed when TWSW data filtering is used with respect to no filtering setup.

Secondly, to observe the effectiveness of proposed ASDDR scheme on same setup, simulation is run separately for both options of ASDDR i.e. once using ASDDR with fixed CHDI and once with variable CHDI. For both these options, at the start of a simulation run, sensing and dissemination intervals are set as: CNSI = 20s, CNDI = 100s and CHDI = 200s. However, as network operation proceeds, as per devised scheme ASDDR dynamically alters CNSI, CNDI and CHDI according to rate of change in event behaviour. The energy saved in this case is computed as the difference in energy consumed by network when ASDDR is used and when no data filtering at all is used and accordingly percentage energy savings are computed.

Figure 4.2 (a) shows comparison in percentage overall network energy saved with varying CNSI's (set statically at each run) when TWSW data filtering is used and also when ASDDR is used once with varying CHDI and once with fixed CHDI. However, adjudging accurate CNSI is very difficult task and is application as well as event behaviour dependent. Percentage energy saved in static sensing scenario ranges from 20.12% at CNSI value of 20 seconds to 40.62% at CNSI value of 5 seconds. Trends clearly indicate that smaller the sensing interval (higher sensing frequency), greater the percentage energy savings achieved with respect to no data filtering case (however total sum of network energies consumed in both cases are higher with higher sensing frequency). This attributes to the fact that with in a cluster temporally closely correlated readings have smaller variability from each other and hence RDF suppresses larger numbers of disseminations causing higher energy savings. More the sensing frequency, more are chances of generating redundant data readings. This does not conclude that one should keep sensing frequency very high for more energy savings, since in absolute terms total sum of network energy consumed is always higher with higher sensing frequency. The requirement of data granularity at application consuming data and estimate of event behaviour (i.e. prediction of rate of change in the value of observed parameters during observation period) together decides optimum sensing frequency for active CNs and is set only one time both for TWSW data filtering and no data filtering case. Therefore, the fact revealed is that if application requirement and estimated event behaviour set CNSI to any of 5s, 10s, 15s or 20s, then how much of percentage of network energy is saved with respect to no data filtering case.

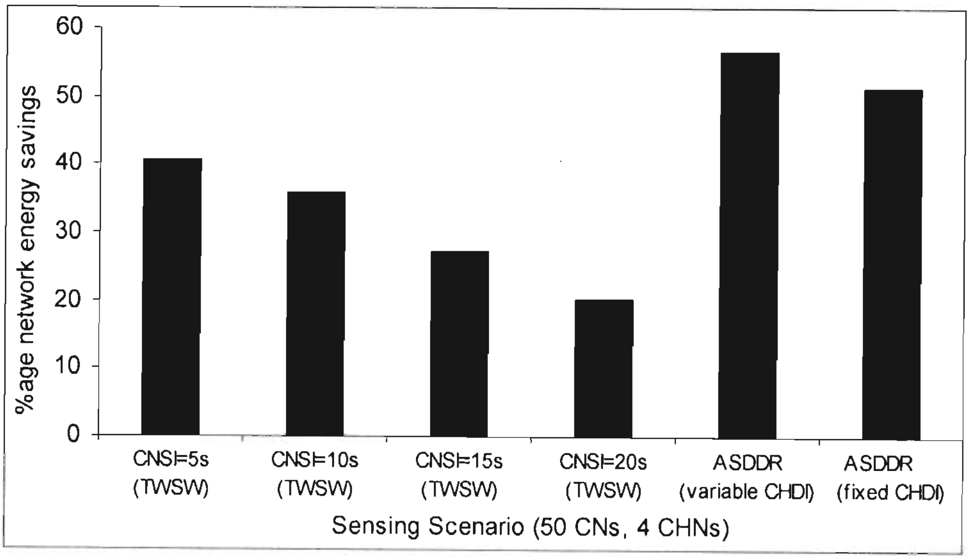


Figure 4.2(a). Effect of different event sensing scenarios on overall network energy consumption with 50 CNs and 4 CHNs

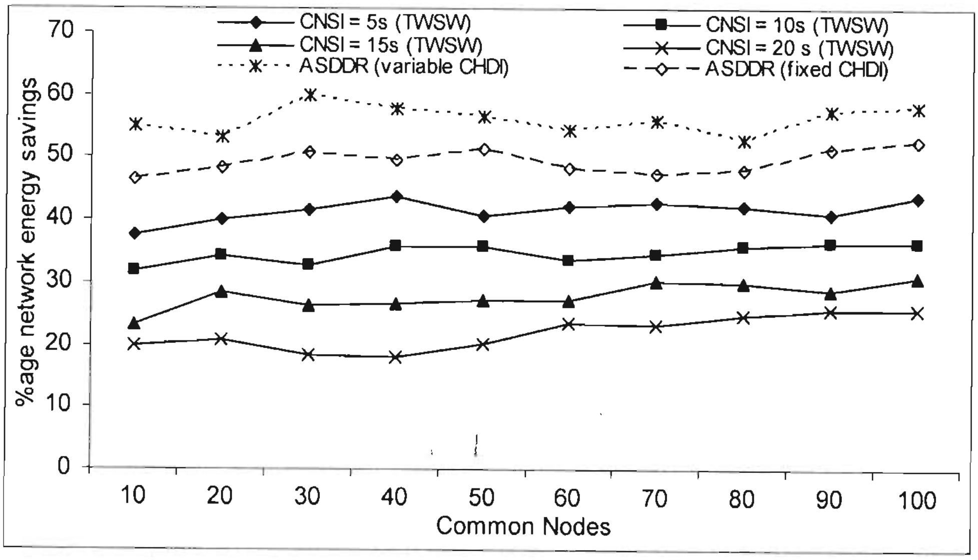


Figure 4.2(b). Effect of CN density (CHNs fixed at 4) and different sensing scenarios on overall network energy consumption

With ADSSR one need not to bother much about exact initial sensing interval as with time it adjusts CNSI dynamically to current event behaviour. As given earlier, here CNSI = 20s, CNDI = 100s and CHDI = 200s are initially set. Figure 4.2(a) also shows percentage energy saved with ASDDR approach (separately for fixed CHDI and variable CHDI) with respect to no data filtering case. ASDDR when used with fixed CHDI achieves 51.4% network energy savings which is significantly higher than respective static sensing scenario with CNSI value of 20seconds which is 20.12%. However, ASDDR when used with variable CHDI achieves further higher network energy savings of 56.7%.

This attributes to the fact that in a fixed CHDI, a CHN has to disseminate data in its buffer periodically after fixed interval. Due to filtering, many a times data in buffer is very less and hence full transmission capacity per dissemination is not utilized which increases cost of per bit transmission.

To further visualize above observations across varying CN density, above simulation runs are repeated with CNs ranging from 10 to 100 CNs. Results in Figure 4.2 (b) reveal that percentage energy savings achieved at CNSI of 5seconds range from 37.7% with 10 CNs to 43.8% with 100 CNs. At other end, percentage energy savings achieved at CNSI of 20 seconds range from 19.8% with 10 CNs to 25.8% with 100 CNs. However, as pointed out earlier also an important point to be noted here is that at higher sensing frequency (smaller CNSI) in absolute terms, total overall energy consumed is more. Actual sensing frequency set depends on the nature of event and/or application requirements.

Figure 4.2 (b) also shows impact of using ASDDR (with fixed CHDI and variable CHDI) as compared to the case when CNSI is set statically and TWSW data filtering is used. Results clearly reveal definite gains when ASDDR is used as compared to scenario where CNSI is set statically for entire observation period. Percentage energy savings achieved using ASDDR when CHDI is fixed ranges from 46.4% with 10 CNs to 52.6% with 100 CNs. This is much higher than savings achieved in static sensing scenario using TWSW only which ranges from 37.7% with 10 CNs to 43.8% with 100 CNs. Further, when varying CHDI is used, this range varies from 55.1% to 58.1% which is still higher than achieved in case of fixed CHDI. The reason for this is already given in this section.

4.3.5 Effect of Event Behaviour (*randomness*) on Overall Network Energy Consumption/Savings

As observed and discussed in previous chapter, degree of *randomness* in event has huge impact on the effectiveness of TWSW data filtering and the general conclusion is that less the *randomness* in event behaviour less is the amount of network energy consumed (more energy savings) due to filtering of redundant reports and higher the *randomness* in event behaviour higher is the network energy consumed (less energy savings). Same fact applies here too to ASDDR with fixed as well as variable CHDI, but the amount and trend over range of *randomness* vary. The results are shown in Figure 4.3(a) and Figure 4.3(b), where Figure 4.3(a) gives overall total amount of network energy

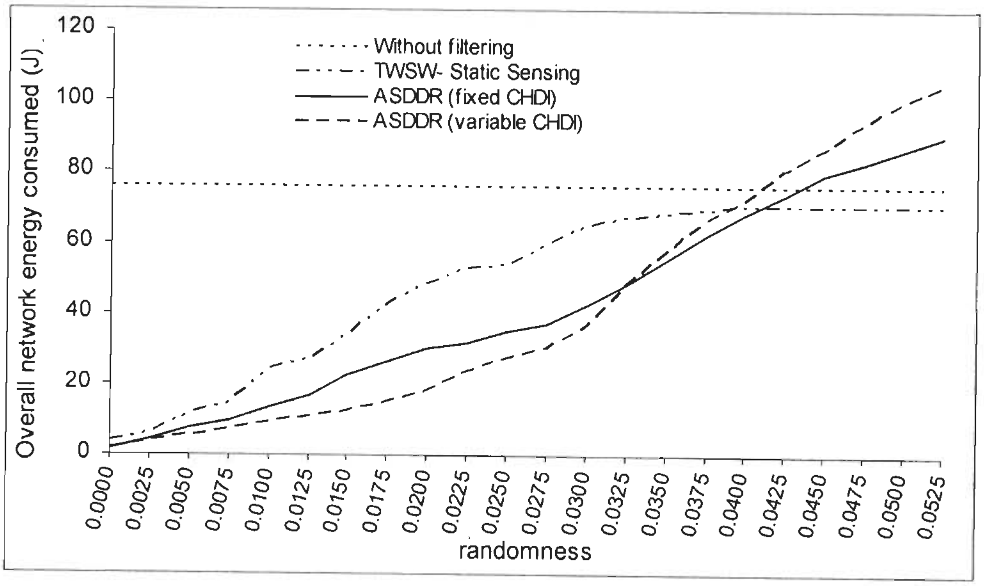


Figure 4.3(a). Effect of event behaviour on overall network energy consumed

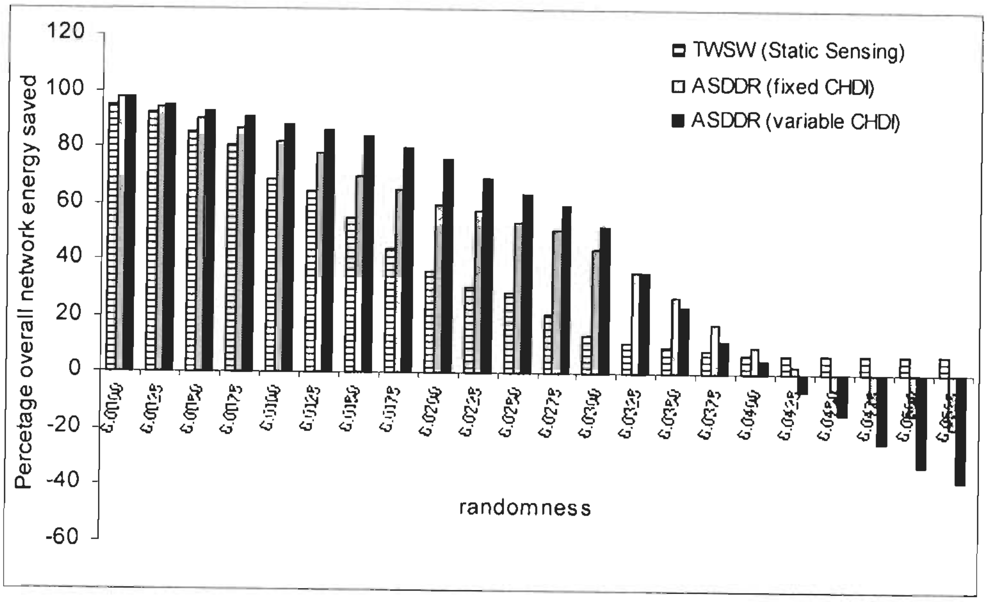


Figure 4.3(b). Effect of event behaviour on percentage of overall network energy savings achieved

consumed whereas for more clarity Figure 4.3(b) gives percentage of overall network energy saved in the form of bar graph.

At very low *randomness*, ASDDR with fixed CHDI gives higher energy savings than static TWSW since former adjusts CNSI and CNDI to adapt to slowly changing event. Thus fewer sensing attempts are made and also CNDI becomes larger resulting in lesser disseminations to CHNs. ASDDR with variable CHDI however yields even more energy savings than ASDDR with fixed CHDI. This is due to the fact that ASDDR with variable CHDI also shortens its CHDI according to shortened CNSI and hence results in to lesser number of disseminations to sink which saves network energy. For instance, at *randomness* 0.02, TWSW with static sensing gives 36.29% network energy savings compared with no filtering case, where as this percentage increases to 60.0% for ASDDR with fixed CHDI and 76.0% for ASDDR with variable CHDI. As *randomness* increases the difference in energy savings achieved between ASDDR and static TWSW starts diminishing rapidly. This clearly is due to the reason that with increased *randomness* ASDDR increases sensing frequency (shortens CNSI) to capture every possible interesting observation where as static TWSW keep sensing frequency same and miss many such interesting moments of observation. ASDDR with variable CHDI even leads to more energy consumptions than ASDDR with fixed CHDI due to shortened CHDI. For instance, at *randomness* value of 0.035 static TWSW achieves 9.71% network energy savings, where as ASDDR with fixed CHDI yields 27.0% and ASDDR with variable CHDI gives 24.0%. At *randomness* 0.04, above percentages are 7.0%, 10.0% and 5.0% respectively.

For *randomness* above 0.04, TWSW with static sensing doesn't give any more energy savings and remain fixed at approximately 7.0%. As has been explained in previous chapter, this is due to the reason that at this *randomness* most of data reports pass RDF filtering and 7.0% energy saving still achieved is only on account of SDF filtering for spurious reports. However, for ASDDR above *randomness* 0.04, the overall network energy consumed becomes more than static TWSW and also even higher than no data filtering case. This is due to reduced value of CNSI (increased sensing frequency), CNDI and also CHDI in case of ASDDR with variable CHDI. For instance in respect of no filtering case, at *randomness* value of 0.05 static TWSW achieves 7.0% network energy savings, where as ASDDR with fixed CHDI consumes 14.0% more network energy and ASDDR with variable CHDI consumes 32.0% more energy. Important point to be noted here is that although no data filtering case as well as static TWSW show lower network energy consumption above *randomness* 0.04, but this is at the cost of missing interesting points of observation which may be necessary for application at sink. Therefore, selection of scheme among these is purely a tradeoff between application requirement and network energy consumption.

4.3.6 Effect of Different Sensing Scenarios on Average Delay

Average delay as defined earlier is the average time taken by a data reading generated at a CN (as a result of event sensing) to reach at sink. Interesting point to note here is that receiver (sink or CHN) assumes previously received value as the current value in case sender (CHN or CN) does not send new data possibly due to data being filtered. To

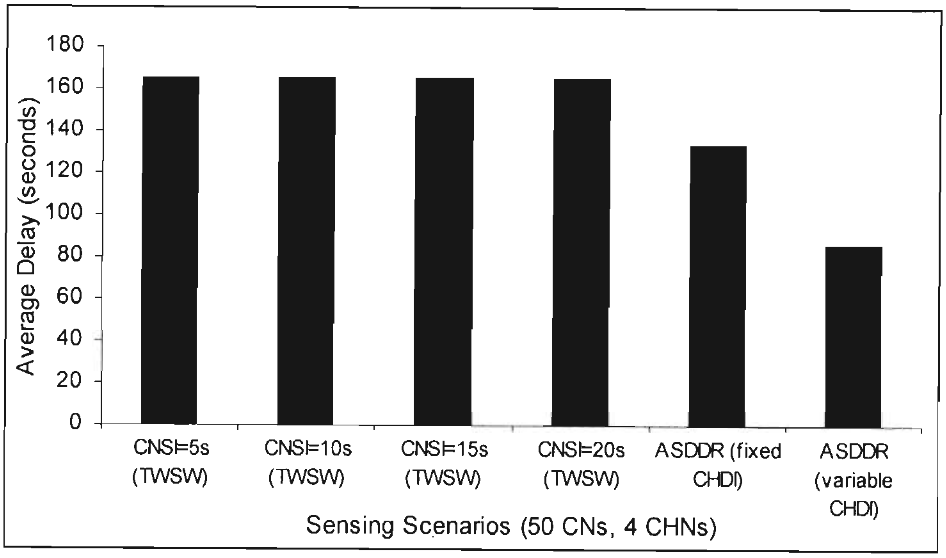


Figure 4.4. Effect of different sensing scenarios on average delay when 50 CNs and 4 CHNs are deployed

complete entry in its buffer, it copies reading from previous slot to new slot for that sender. Therefore, in current study from information transfer point of view, we consider data reading being forwarded to receiver even if it is blocked at sender as redundant. In any case, at the end data reading is conveyed to sink. Also, all CNs although have same CNSI and CNDI, still they sense and disseminate during different times due to lack of global time synchronisation as prevalent in most distributed systems. For, example, a CHN listens to disseminations for 200 seconds from CNs in its cluster and receives data reports at different times spread across 200 seconds.

In case of static sensing as used in TWSW data filtering, each of CNSI (20 seconds), CNDI (100 seconds) and CHDI (200 seconds) once set remain unchanged during entire observation period. The average delay as revealed in Figure 4.4 is 165 seconds. However, to observe the effect of different sensing frequencies on average delay, different CNSI values are set manually for different runs while CNDI and CHDI are kept fixed. The only difference with changed value of CNSI lies in the number of data readings accumulated in CN_{buffer} during a complete CNDI, where as they are disseminated after 100 seconds to CHN and CHN disseminates to sink after 200 seconds. Therefore, average delay remains same at 165 seconds even for different CNSIs when no frequency adaptation is used. Results in Figure 4.4 further reveal that average delay achieved by ASDDR with fixed CHDI is smaller than static sensing and is 133.6 seconds. Also, ASDDR with variable CHDI gives even lesser average delay of 86.35 seconds than ASDDR with fixed CHDI.

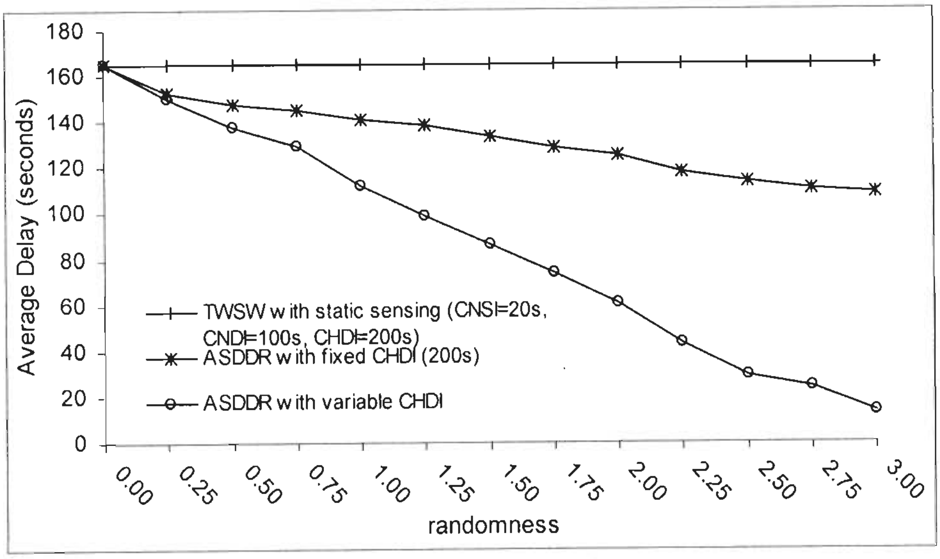


Figure 4.5. Effect of event behaviour on average delay when static sensing, ASDDR with fixed CHDI and ASDR with variable CHDI is used

4.3.7 Effect of Event Behaviour (randomness) on Average Delay

In scenario, when sensing frequency is set statically, all of CNSI (20 seconds), CNDI (100 seconds) and CHDI (200 seconds) remain unchanged during entire observation period. Therefore, *randomness* in event behaviour does not have any impact on average delay. Average delay in this case remains around 165 seconds. In ASDDR, when sensing frequency is set dynamically, event behaviour has interesting impact on average delay. At the start of observation period sensing and dissemination intervals are set as above and then are set dynamically by CHNs and CNs. As shown in Figure 4.5, ASDDR with fixed CHDI as well as with variable CHDI gives much less average delay than in case of static sensing scenario. For instance in ASDDR with fixed CHDI, at zero *randomness* average delay is same as in static sensing, at *randomness* 1.5 it gives 19.05% less average delay than static sensing and for highly random event with *randomness* 3.0 the difference is 34.26%. This is due to the reason that with increased *randomness*, CNSI and consequently CNDI are decreased resulting data reading to take lesser time waiting in CN_{buffer} to get disseminated to CHN. *Randomness* above 3.0 results in blocking entire data reports treated as spurious due to β_{CN} and β_{CH} being 3 °C which maps to *randomness* 3.0, hence results not plotted beyond this. For ASDDR with variable CHDI, average delay further decreases due to shortened CHDI with more *randomness* in event behaviour. Thus, data reading takes lesser time in CN_{buffer} as well as in CH_{buffer} waiting to be disseminated as compared to static sensing. For instance at zero *randomness*, average delay is same as in static sensing as well as in ASDDR with fixed CHDI, however at *randomness* 1.5 it gives

47.67% less average delay than static sensing and for highly random event with *randomness* 3.0 the difference is 91.8%.

4.4 Conclusions

For predictable scenarios, TWSW data filtering with static sensing successfully filters undesired data which is either redundant or spurious to application at sink. However, in most scenarios the rate of change in event behaviour is unpredictable and can not be adjudged at the very beginning while deciding sensing frequency for CNs. Thus, improper setting of sensing frequency either leads to wastage of precious network energy or misses interesting points of observation leading to loss of accuracy in monitoring and control. ASDDR however is an approach that solves this problem by adapting sensing and dissemination frequencies to changing behaviour of an event on the fly without the need for manually changing it.

ASDDR approach is very simple and easy to implement although very effective as revealed by simulation study. For relatively less random behaviour of an event, ASDDR (both with fixed CHDI and with varying CHDI) gives best performance in terms of overall network energy consumption and at the same time transfers same information to sink with less data as compared to no data filtering or TWSW data filtering. At very high rate of change in event behaviour (*randomness*) however it may consume more network energy than either of no data filtering or TWSW data filtering but it captures every possible interesting observation about event. Saving network energy at the cost of lost accuracy or wrong inference is seriously damaging and unwanted situation for most applications.

Another important usefulness of ASDDR scheme is that average delay reduces with increased *randomness* which is quite interesting. With more *randomness* more observations about event are generated at CNs which must be sent to sink. It seems that longer data queues will be formed at CNs and CHNs leading to more delays. ASDDR however uses a dissemination model which responds to changed sensing frequency by adjusting dissemination intervals at CNs and CHNs in such a manner that average delay gets reduced. Application may impose certain upper and lower limit on the reporting time for CHNs and between these limits ASDDR can easily adjust sensing and dissemination frequencies.

Chapter 5

Dual Radio based Grid Construction and Data Dissemination

5.1 Introduction

Various energy efficient data dissemination methods have been proposed over the years to reduce energy consumption in WSNs. The network topology used underneath hugely affects the performance of dissemination approach in terms of network overheads, delays and energy consumption etc. Based on network topology, various data routing schemes in WSNs can be categorized into two groups: flat and hierarchical. In flat routing schemes, no clustering is used and all query/data flow multi-hop from node to node with all nodes treated at same level. Directed Diffusion [17], SPIN [143] [65], Gradient Based Routing [18], EAR [116], MCFA [42] and Rumor Routing [25] are few among many routing schemes falling under flat category.

Hierarchical approaches convert entire sensor field into collection of small areas with nodes in each area forming a group or cluster. Organizing nodes in a WSN into groups/clusters provide some degree of modularity for network management by performing coordinated activities with in a group/cluster. As discussed in literature survey, clustering also helps in restricting flooding with in a cluster and hides topological details along with providing data aggregation points at cluster heads. TTDD [53], LEACH [148], PEGASIS [120], TEEN & APTEEN [4] and VGA [67] are few schemes representing this category.

In the absence of node ids, forming clusters in WSNs is however not simple and many factors unique to WSN make it very complex task to accomplish. Few of these factors and their implications are:

i. Once nodes are randomly deployed, the unattended nature of WSN puts itself in an auto-configuration mode. The obvious questions to be answered in this mode are:
   a. When to start cluster formation? If cluster formation doesn't start in time, path setup may be delayed causing loss of useful information.
   b. Once decided to form clusters, who will initiate the process and which nodes shall act as cluster heads? This may need some sort of coordination among nodes.
   c. What is clustering criteria? This includes deciding about the shape/size of a cluster, cluster head selection and assigning nodes' membership to cluster heads.
   d. How the clustering process halts to avoid repetitive cluster formation over same region again and again?
   e. How long should clusters be maintained in the absence of occurrence of any event?

ii. Generally in WSNs, cluster heads provide points for aggregating data from various nodes in their respective clusters and form a backbone to move data towards sink i.e. they communicate among themselves and with sink either directly or in multi-hop using on-board radios. Apart from this, cluster heads also indulge in cluster management activities like broadcasting about their presence in their local clustering range, hearing to member nodes and forming indexed list of member nodes and updating its database regularly by including any new node joining its cluster etc. Therefore, CHNs consume much more energy than other nodes do. Since, a cluster is vulnerable to single point of failure at CHN, a poor clustering strategy may cause multiple CHNs to exhaust their batteries completely and cause WSN to partition in to unusable segments. Therefore, while employing a clustering criteria, following design questions arise at cluster heads:
   a. How to minimize the energy loss at CHNs? Is it possible to rotate the role of cluster heads among some nodes?
   b. Is it possible to separate out activities at cluster heads into different categories where activity in each category has different power requirements? If yes, is it possible to use different power modes of SNs for different set of activities at CHNs? Future SNs are expected to have multiple or adjustable power modes for communication and processing (some presently available SNs already have).
   c. SNs are highly vulnerable to failures due to energy drain, physical damage due to environmental conditions, hardware/software malfunction etc. Can clustering

provide some tolerance against these failures at cluster head level to improve single-point and single-time failure situation?

Apart from answering above clustering design questions, mobility of sink and event pose major challenge while developing a data dissemination scheme for WSN that uses some form of clustering. Multiplicity of sinks and events further complicates the path set up and data dissemination in a WSN. Various scenarios that emerge due to mobility and multiplicity of sinks/events in a WSN are:

i. Sink/event moves with in a region that is part of current cluster.
ii. Sink/event slowly moves to different location that falls under different cluster.
iii. Another sink/event appears at some different region of the network which falls under separate cluster.
iv. Another sink/event appears in the same cluster.

Keeping above design questions in mind, we propose Dual Radio based Grid Construction and Data Dissemination scheme which tries to answer many (if not all) of above design questions [136]. Proposed scheme exploits dual radio modes of a SN to form a virtual grid across sensor field. Virtual grid helps in defining clusters, setting path between source-sink pairs and handling movement as well as multiplicity of sinks/events effectively. Based upon this grid and clustering, we develop methods for handling multiplicity and movements of sinks and events in the sensor field so as to ensure continuous data delivery from a source node to sink.

5.2 Two-Tier Data Dissemination Approach for WSNs

Before giving proposed approach, we first elaborate on most relevant Two-Tier Data Dissemination (TTDD) grid based data dissemination approach which constructs virtual grid over entire sensor field. Although, TTDD has also been discussed in chapter 2, but here we elaborate more on details of grid construction and path setup to differentiate it from proposed scheme.

TTDD approach addresses the multiple and mobile sink problem in WSN. Instead of propagating query messages from each sink to all the sensors to update data forwarding information, TTDD uses a grid structure so that only sensors located at grid points need to acquire the forwarding information. Upon detection of an event, instead of passively waiting for data queries from sinks, the data source proactively builds a grid structure throughout the sensor field and sets up the forwarding information at the sensors closest to grid points (called dissemination nodes).

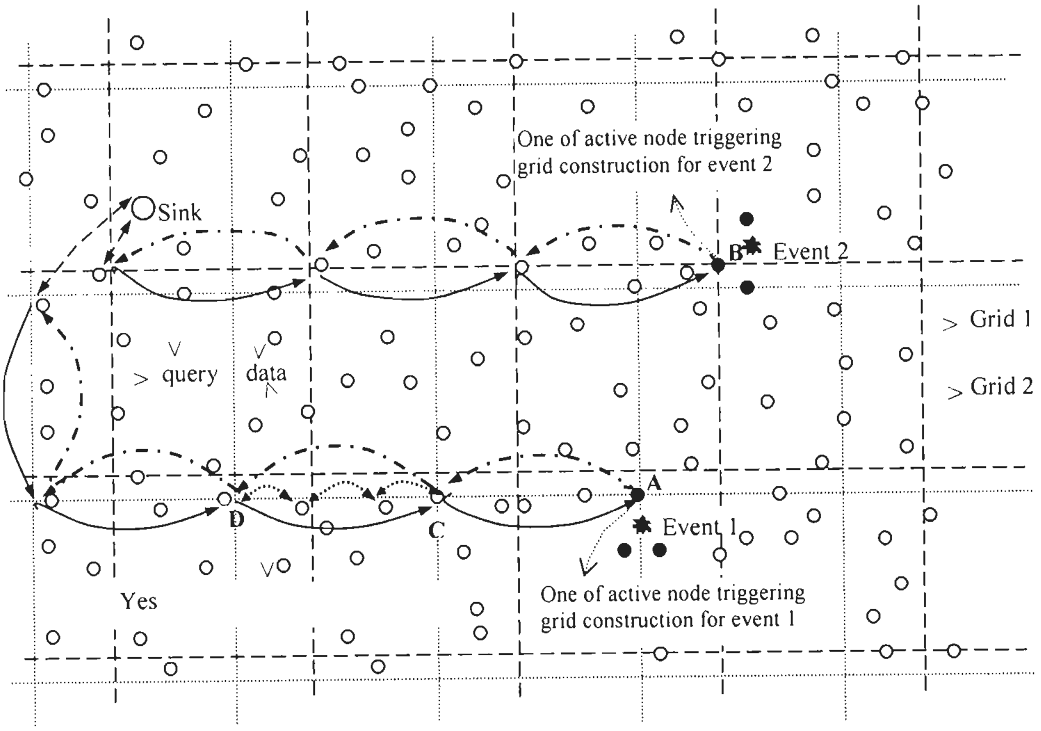


Figure 5.1. Grid formation and query/data flow from source to sink in TTDD

With this grid structure in place, as shown in Figure 5.1, a query from a sink traverses two tiers to reach a source. The lower tier is within the local grid square of the sink's current location (called cells), and the higher tier is made of the dissemination nodes on the grid. The sink floods its query within a cell. When the nearest dissemination node for the requested data receives the query, it forwards the query to its upstream dissemination node toward the source, which in turns further forwards the query, until it reaches either the source or a dissemination node that is already receiving data from the source (e.g., upon requests from other sinks). This query forwarding process provides the information of the path to the sink, to enable data from the source to traverse the same two tiers as the query but in the reverse order.

TTDD make following assumptions:

- SNs are homogeneous and stationary.
- SNs communicate with each other through short-range radios. Long-range data delivery is accomplished by forwarding data across multiple hops.
- Each SN knows its geographical coordinates which is used to tag sensed data and helps in constructing and maintaining grid.
- There can be multiple slowly moving sinks at any time in sensor field and the number may vary with time.

With a grid structure for each data source, queries from multiple mobile sinks are confined within their local cells only, thus avoiding excessive energy consumption and network overload from global flooding by multiple sinks. When a sink moves more than a cell size away from its previous location, it performs another local flooding of data query which will reach a new dissemination node. Along its way toward the source, this query will stop at a dissemination node that is already receiving data from the source. This dissemination node then forwards data downstream towards the sink. This way, even when sinks move continuously, higher-tier data forwarding changes incrementally and the sinks can receive data without interruption. Furthermore, because only those sensors on the grid points (serving as dissemination nodes) participate in data dissemination, other sensors are relieved from maintaining states. TTDD thus scales to a large number of sources/sinks.

Figure 5.1 shows grid formation and query/data path set up. Two different events appearing in sensor field triggers construction of two separate grids at active nodes A and B. For clarity of figure, transmission between adjoining dissemination nodes is shown direct, whereas it is actually multihop as shown in figure for pair C - D.

The major drawback of TTDD is that every time a new event appears in sensor field, a new grid is constructed. This consumes significant amount of energy and costs major overheads if number of events are more.

Next we give our proposed dual radio based grid construction and path set up strategy for homogeneous WSN, known as Grid Based Data Dissemination (GBDD). Table 5.1 summarizes notations and symbols used in next few sections.

5.3 Grid Based Data Dissemination (GBDD)

5.3.1 Objectives

After going through various clustering/data dissemination design challenges in WSNs (Section 5.1) and closely related TTDD approach, to achieve larger objective of energy efficiency the immediate objectives of our proposed approach are set as:

- To organize randomly deployed SNs in a sensor filed into clusters of suitable size by forming a virtual grid over entire sensor field.
- To exploit dual radio mode of a SN to decide size of the cell of a grid.
- Identify and separate out activities into two groups at CHNs (later also called dissemination nodes) so that different power modes of a SN can be used for activities from different group.

- To handle the movement and multiplicity of sink(s) and event(s) for uninterrupted data delivery.
- To reduce the overhead incurred in TTDD by eliminating the need to construct separate grid for each different event appearing in sensor field event when a valid grid is present.
- To develop a grid structure which is tailor made for further proposed cooperative caching.

5.3.2 Assumptions and System Environment

We make following assumptions in line with the technological trends and existing work:

- The sensor field is flat, spanning in two dimensional plane.
- All SNs are homogeneous capable of communicating with each other through radio signals using omni-directional antennas. All nodes normally communicate using high power radio, however cluster heads use low power radio while communicating with nodes in a restricted Alternate Dissemination Zone (ADZ) around it (to be explained next). The communication between two nodes is symmetric (i.e. if I can hear you, you can here me too).

Table 5.1. Symbols and Notations Used in GBDD

Symbol	**Definition**
R_H	Radio transmission range of a node in high power mode
R_L	Radio transmission range of a node in low power mode
α	Size of the square shaped cell of a grid
CP	Calculated actual crossing point (coordinates) of a cell of a virtual grid
DN	A node closest to a CP among all its neighbors and is known as Dissemination Node
IDN	First Dissemination Node towards sink and is known as Immediate Dissemination Node
d	Diagonal of a cell of a grid and is set to R_H - R_L
D_L	Distance from a closest node found using geographic greedy forwarding to corresponding CP
ADZ	A region of radius $R_L/2$ around a DN from where alternate DN is appointed in case of failure of existing DN and is called as Alternate Dissemination Zone
SDN	First DN towards an event to whom active SNs (which detected event) send sensed readings and is known as Source Dissemination Node
M_{setup}	Path setup message sent to SDN by active nodes for initiating path set process

- SNs can sense multiple parameters like heat, light, humidity, pressure etc and digitize the signal before further dissemination.
- Once deployed, SNs do not change their location, however sink and event may move slowly i.e. and thus there is no guarantee that sink/event location at time t and time $t + \tau$ is same.
- Each SN is assumed to be aware of its own location i.e. it knows its geographical coordinates (x, y) which serve as node identifier. Since we assume SNs stationary, assigning coordinates to them is one time job. Node localization is now well researched area in WSN design and recently many localization techniques have been proposed which do not use onboard GPS on nodes [95] [48], which otherwise might consume a lot of node's energy and also make it cost-ineffective. In conventional wireless networks geo-location estimation based on received signal strength has been used for long but suffers from its own disadvantages like errors due to multipath fading. At the same time there have been efforts to overcome this problem. For example, Grover *et al.* [104] exploit spatial diversity combined with channel knowledge at the receiver to combat fading effects to increase accuracy in location estimation.
- Each node is capable of communicating in dual mode i.e. in low power radio mode and high power radio mode.

The assumption of dual radio mode of SNs is in line with recent trends in SN development. Chapter 2 reviews research in the area of power adjusting modes of SNs, multi radio usage, reconfigurable power management in SNs and adaptive power/radio control.

5.3.3 Dual Radio Based Grid Construction

Starting Grid Construction Process

We propose a grid construction scheme where size of the cell is determined by dual radio mode capability of a SN. Unlike TTDD, where adjoining dissemination nodes (nodes representing corners of a cell) communicate using multihop transmissions, in this approach cell size is set in a manner where they communicate directly using high power radio. Also, all nodes with in a cell can communicate directly with any of its four corner nodes. In TTDD, appearance of an event and subsequent detection by some active nodes initiates grid construction. But in proposed approach first sink when interested in some data first time starts grid construction process by keeping itself at one of crossing point (CP) of the grid if already a valid grid is not present. The two dimensional geographical coordinates (x,y) of this sink thus become starting point for formation of grid of square sized cells. Whereas, any sink appearing during a period when a valid grid exists, shares existing grid and thus obviate the need to construct a new grid. Both short and long radio ranges of a dual radio decides the cell size and all nodes with in a cell form a cluster with one of corner nodes of the cell acting as cluster head.

Let R_H and R_L be the transmission ranges of every SN while working in high power radio mode and low power radio mode respectively. The size of the square sized cell (α) of the grid is set as:

$\alpha = d/\sqrt{2} = (R_H - R_L)/\sqrt{2}$, where d is the diagonal of a cell. (5.1)

The reason behind setting d to $R_H - R_L$ is given next in section defining cell size and ADZ.

All other CPs (x_i,y_i) are calculated from sink(x,y) as:

$\{x_i = x + i \cdot \alpha,\ y_i = y + j \cdot \alpha;\ i,j = \pm 1, \pm 2, \pm 3, \ldots\ldots\ldots\}$ (5.2)

Given its own coordinates (x,y) and α as above, sink calculates coordinates of four adjoining CPs. Since α is much smaller than R_H, sink can transmit a signal in single hop up to any of CPs, but SN may not be exactly there at the calculated coordinate (x_i,y_i) of an CP. Therefore, a node nearest to CP is found and is designated as dissemination node (DN). Although CP is one hop apart from sink, this is achieved by following a geographic greedy forwarding method [167]. In geographic greedy forwarding, a node sends message to its neighbor with smallest distance to destined CP. The neighbor node continues forwarding message to its neighbor that has minimum distance to CP and thus on each hop a node that has least distance to CP among all its peers (sender's neighbors) is selected. Here sink forwards grid formation message with coordinates of respective CPs in each of four directions. Finally, grid formation message stops at a node that has least distance to CP among all its neighbors. However, if distance D_L of this node from CP is less than or equal to half of low power radio range (i.e. $D_L <= R_L/2$), it is finalized as DN, otherwise node simply drops the message. This condition helps in terminating the grid formation process at the boundaries of sensor field.

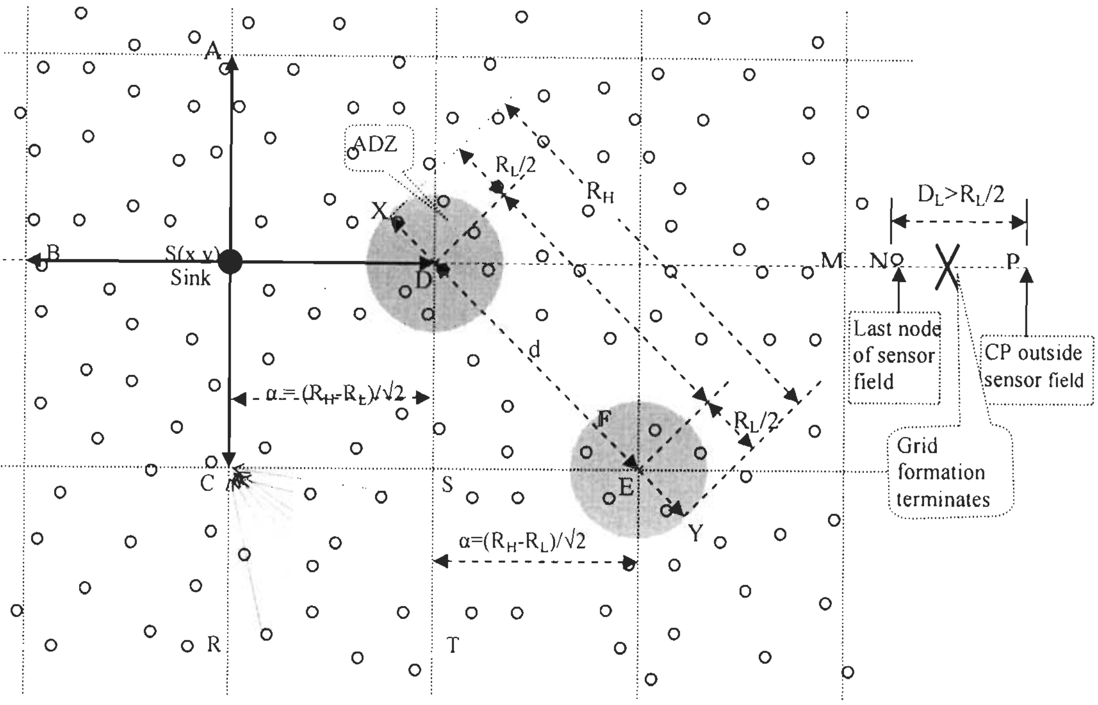


Figure 5.2. Grid formation using dual mode radio

Terminating Grid Construction Process

Once started, grid construction process must terminate automatically as soon as virtual grid is constructed by appointing all DNs. The condition, {if $D_L<=R_l/2$, designate node as DN corresponding to current CP, otherwise drop grid formation message with out forwarding it any further} helps in making decision that the CP for which DN is to appointed is well outside the sensor field. In other way, grid construction has reached at the end of this side of the sensor field. For example in Figure 5.2, P is new CP calculated from M and N is the last node on that side of the boundary of sensor field. M sends grid formation message to its neighbor N, but it has $D_L>R_L/2$ and has no other neighbor which is closer to P. Hence, grid formation on this side of the sensor field terminates here. Reason behind choosing distance limit as $R_l/2$ in above condition is given in next section defining cell size and alternate DN zone.

New DNs further start similar grid formation process until their neighboring DNs are selected or process damps completely at boundaries. Flowchart in Figure 5.3 explains grid construction process.

5.3.4 Defining Cell Size and Alternate DN Zone

Cell size is decided by taking high power radio range (R_H), as well as low power radio range (R_L) of a SN. A simple clustering scheme is used where all nodes in a cell

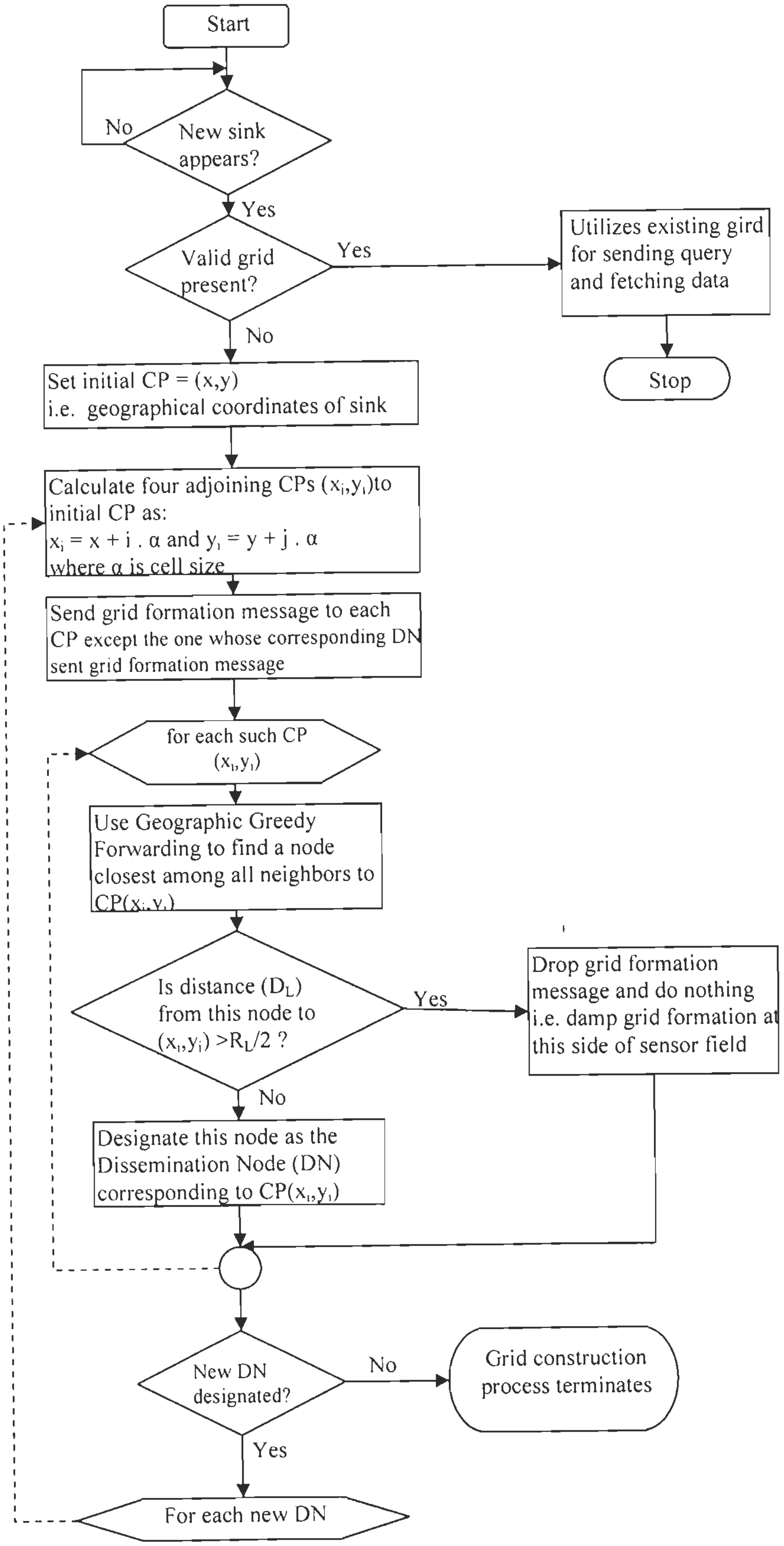


Figure 5.3. Flowchart for GBDD Grid Construction

must be able to communicate in single hop using R_H with its cluster head as well as with each other. If two nodes are at two extremes of a diagonal of a square sized cell, then maximum size of diagonal must not exceed R_H. We use nodes falling in a circular region of radius $R_L/2$ around DN (as shown by shaded region in Figure 5.2 around nodes D and E) to form a set which can be used for selecting an alternate DN in case present DN fails or its energy level falls below a limit. This zone is named as alternate dissemination zone (ADZ). Energy level limit is based on factors like load balancing which is used to avoid draining energy of certain nodes completely while other nodes are rarely used or situation where present DN has failed due to some other reasons like crash, theft, hardware/software failure etc.

In either case, present DN is substituted by a node from its ADZ. All nodes in a ADZ must communicate in single hop with each other using low power radio R_L for any zone coordination activities like new DN election, energy level indexing of nodes in zone etc. Therefore, in order for two nodes at each end of a diameter of ADZ to communicate in single hop using low power radio, it must not exceed R_L i.e. radius must not exceed $R_L/2$. Also, DN communicates with nodes in ADZ using low power radio. On failure of present DN, say E in Figure 5.2, it is replaced by an alternate DN on boundary of ADZ, say at Y. A node in the same zone is at opposite end of diameter on the boundary; say at F in Figure 5.2. For new DN at Y to communicate in single hop with zone node at F, the diameter of ADZ must be less than or equal to R_L.

Also, in case of failure of a DN D in Figure 5.2, it is replaced by an alternate DN in its ADZ and this node may be at X i.e. diagonally opposite extreme of Y. Therefore, distance X-Y is the maximum distance created between two DNs caused by failure of present DNs. Since, R_H is the maximum distance up to which DN can communicate in single hop, it is clear that distance X-Y must not exceed R_H if two DNs are to remain in one hop communication range. Accordingly, diagonal d of the square sized cell must be set at maximum as:

$$d = R_H - R_L$$

and thus side of cell is:

$$\alpha = d/\sqrt{2} = (R_H - R_L)/\sqrt{2} \tag{5.3}$$

5.3.5 Clustering

A cell of a grid is treated as one cluster with one of its four corner DNs selected as cluster head. Each SN in a cluster communicates directly in single hop with its cluster head. The selection of cluster head among its corner DNs is based on the direction of the sink who initiated the grid formation process. SN in a cell receives grid formation message from every surrounding DN during geographic greedy forwarding process of grid construction. This message includes the coordinates of the new CP, coordinates of sink, as well as sender DN's coordinates. This enables nodes to know the direction of the sink with respect to its own coordinates. Thus, SNs in a cell select DN with minimum geographical distance to sink as their cluster head. For example, in Figure 5.2, nodes in cell C-S-R-T select C as their cluster head, since it is closest to sink compared to S, R or T.

5.3.6 Handling DN Failure

While forming ADZ around DN, every node within a region of radius $R_L/2$ with DN at center broadcasts a tuple comprising of its coordinates (id) and its residue energy using low power radio. Hence, each node in zone hears a small tuple from every other node in ADZ. Accordingly, each node in ADZ including DN creates an indexed list of node-ids in descending order of residue energy. Each time an alternate DN is to be selected, it selects a node from indexed list in sequence from top i.e. node with highest residue energy. Proposed grid construction mechanism is such that irrespective of the location of new DN in ADZ, it always will remain in one hop transmission range from all its neighboring DNs and thus listens to their transmissions with out any change.

Point to be noted here is that the ADZ once formed remain static and do not change with the selection of alternate DN. If new DN is made center of zone by forming new ADZ, symmetry of grid/DNs will be disturbed and situation may arise where adjoining zones drifts away from each other such that their DNs are not able to communicate directly in single hop. Also, new DN already posses indexed list of nodes in ADZ as above and it simply deletes the entry of the DN from which it has taken over.

5.3.7 Sensing Scenario and Path Setup

Algorithm 5.1 gives complete sensing scenario and path setup procedure deployed and is elaborated diagrammatically in Figure 5.4. Each SN can sense multiple parameters about the event depending on the type of query received from its cluster head DN. However, immediately after grid formation, SNs which have event in their sensing range send path set up message M_{setup} to their cluster head DN called Source Dissemination Node (SDN). If event is not yet present, no path setup takes place and this also prevents sink from unnecessarily issuing queries. Once event appears, path setup message M_{setup} is sent by few SNs to corresponding SDN as above. SDN forwards M_{setup} to its upstream DN

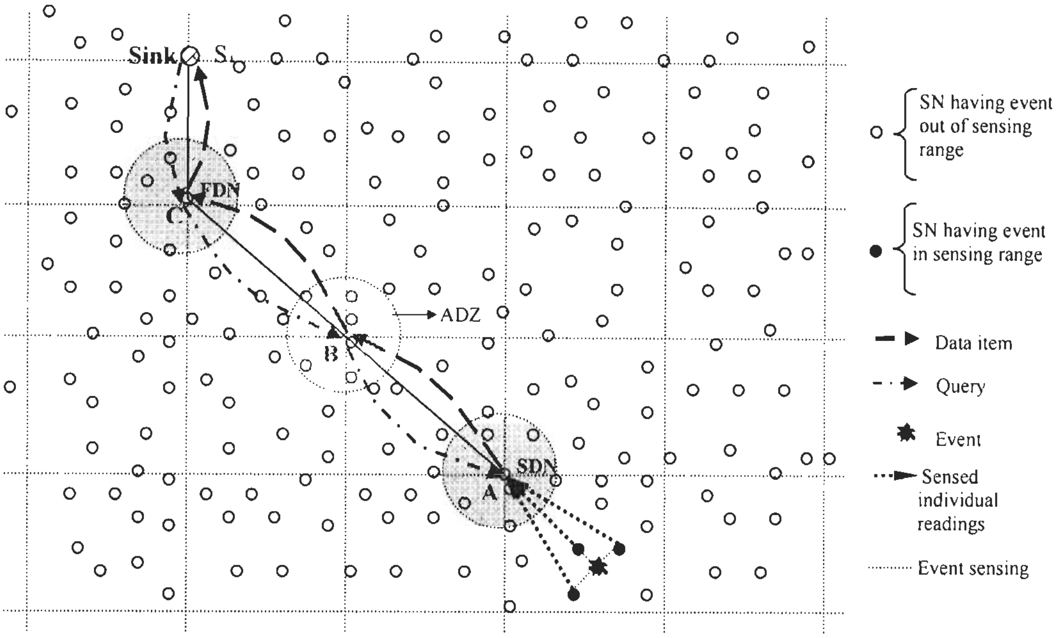


Figure 5.4. Setting Query and Data Flow Path

i.e, DN towards sink. Note that all DNs gather information about upstream neighbor during grid formation process (i.e. out of its entire one hop DNs, it selects the one which is nearest to sink). Upstream DN also forwards it to its upstream DN and so on till M_{setup} reaches at sink. This sets up initial path for query and data flow. All subsequent data retrievals are demand based scenario as used in this work.

After initial path set up, query is issued by the sink for required data item which flows DN to DN till it reaches a SDN. On receiving query message regarding certain type of data from SDN, each SN in the vicinity of the event senses the desired parameter and sends individual reading to its cluster head i.e. SDN. SDN aggregates readings to infer actual data item and forwards it towards sink.

5.4 Handling Multiple and Mobile Sinks/Events

5.4.1 Handling Multiple Events and their Movements

When event location changes with in a cell, the only change that occurs is in the nodes that sense event i.e. for some nodes event may go out of range and for few others it may come in their sensing range. Nodes having event in their sensing range become active (i.e. they wake up) and all other nodes remain in sleep mode. While SDN is to fetch data from sensing nodes, it broadcasts message in the cell for which it is a cluster head and all active nodes respond to it by sensing the required parameter and transmitting it back to SDN. Therefore, event movement with in cell is automatically handled. When event

Algorithm 5.1. Setting query/data path between event and sink

```
1. until (Event is not Detected)
       i.  SNs remain passive but awakened to observe the appearance of an event
       ii. No path setup is initiated        // Unnecessary path setup avoided
2. for (each SN having event in its sensing range)   //Event appears
       i.  Activate itself to sense the event
       ii. Send $M_{setup}$ (a path setup message) to its cluster head called SDN
3. if (this is not the first $M_{setup}$ message received at SDN from any of nodes from current cell
   and a path between SDN-sink pair already exists)
       i.  Send acknowledgement message conveying existence of path and do nothing
               // Path has already been set
4. else
       repeat {
       i.   Using high power radio, DN (SDN in first step) broadcasts $M_{setup}$ message
            comprising GPS coordinates of its upstream DN towards sink (about which it
            has learned during virtual grid formation) and also its own coordinates.
       ii.  Neighboring DN having coordinates as in $M_{setup}$ only receives this message and
            others ignore it.
       iii. Upstream DN sets pointer towards downstream DN by remembering its
            coordinates also found in $M_{setup}$. }
       until (sink is discovered)
5. Send acknowledgement of path setup from sink to SDN following downstream pointers
   at each DN including sink.
```

crosses boundary of a cell and comes in the sensing range of a node or nodes in new cell, they become active and a path setup procedure is initiated. Active nodes immediately transmit path setup message M_{setup} to their CHN which now becomes new SDN for the event in this cell. M_{setup} includes current values of all programmed parameters about the event. These parameters help intersecting node (which may be the old SDN) of new and existing path to decide whether event is new or old event moved to new cell. If event moves to a cell whose one of corner nodes is old SDN or DN on existing path and no other corner node is on data/query path, old SDN or DN on existing path is used as new SDN for event in this cell. Even if this is new event in this cell, old SDN or DN on existing path serves as SDN for both events.

Figure 5.5 shows initial scenario when event is in cell a and its SDN is node S. As shown in Figure 5.6, if event moves to any of cells a, b or d, its sensing nodes still use old SDN S as their new SDN. Cell m has DN A on existing path as its corner node and if event moves to this cell, SNs sensing event selects A as new SDN. Node A (and also every

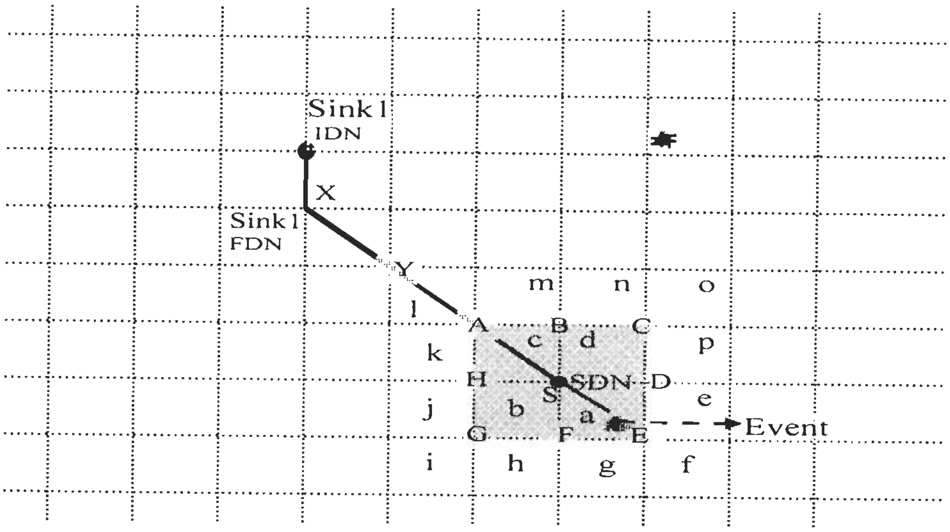


Figure 5.5. Initial scenario when event is in cell a and its SDN is node S

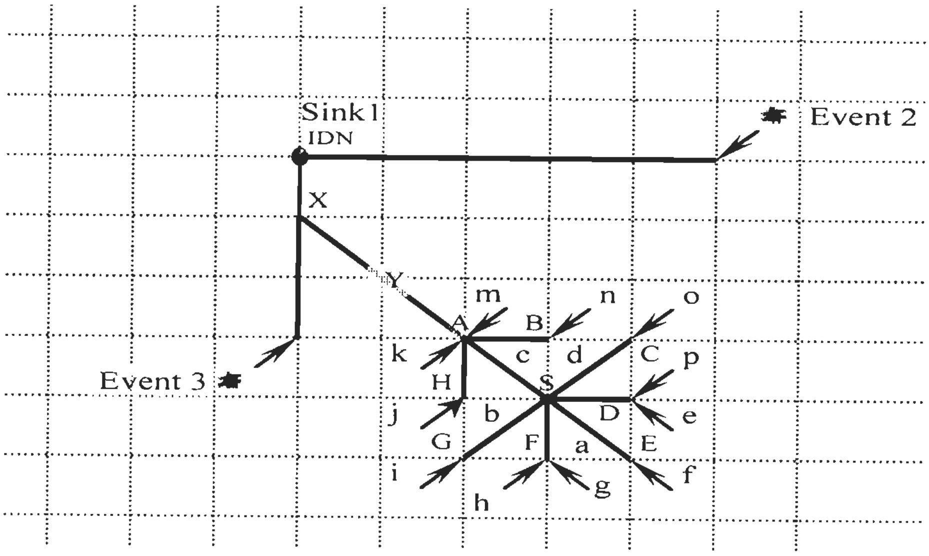


Figure 5.6. Event moves to any of cells a, b or d, but sensing nodes still use old SDN S as their new SDN

intersecting node) also decides whether event is new or old event shifted to this cell. Initial parameter readings in path setup message reveal this for node A. In case event is new, it keeps portion of existing path from itself to old SDN intact by retaining pointer to downstream node S, otherwise if event is not new, but existing event has moved to cell m, DN A removes pointer to downstream node S. Likewise, all intersecting DNs decide whether to retain portion of old path from itself to old SDN or not.

If event moves to a cell or new event appears in a cell whose one of corner nodes is old SDN and one or more of other corner nodes are also DNs on existing path, then a DN on existing path closest to sink is selected as new SDN. For example in Figure 5.6, cell c selects node A as new SDN instead of old SDN S. If event moves to a cell none of whose corner nodes are either old SDN or DN on existing path, then all corner DNs of this cell broadcast probe message to their respective one hop neighbor DNs only. One of the corner DNs of this cell is guaranteed to find old SDN or some other DN on existing path as its one hop neighbor. Cell selects this corner DN as its new SDN which connects to detected old SDN or a DN on existing path to complete new path. In Figure 5.5, cells e, f, g, h, i, o and p selects corner DNs D, E, F, F, G, C and D respectively their new SDN in case event moves to any of these cells. Each one of these new SDNs has old SDN S as one hop neighbor which becomes one hop upstream DN for new SDN to complete path. Whereas cells n and j selects corner DNs B and H respectively their new SDN since both B and H have old SDN S and DN A on existing path as one hop neighbors. However, they make DN A as upstream node to complete path as it is closer to sink as compared to old SDN S.

If new event appears in a cell other than above surrounding cells, similar procedure takes place for path setup as in first event case, with the difference that if new path and some existing path intersect at a node other than sink, new path setup stops at a intersecting DN. At intersecting DN, downstream DN pointer to existing path is also retained. For example in Figure 5.6 at DN X, pointer to downstream DN Y is retained. Path from sink up to point of intersection is shared by two SDNs. Thus, multiple SDNs exist in case of multiple events and paths converge at points of intersection. Path set up message M_{setup} carries sufficient information enabling intersecting node to decide whether event is new or previously existing event shifted to adjoining cell.

5.4.2 Handling Multiplicity of Sinks and their Movements

First sink initiates grid construction by taking its coordinates as first CP and accordingly other CPs of grid are calculated. In case of sink movement, a DN closest to its initial CP is elected as Immediate Dissemination Node (IDN). IDN takes over the responsibility of receiving query from sink and communicating data to it. While sink moves in any of four cells around CP, IDN can communicate with it in single hop and hence no extra path maintenance is required. However, out of these four cells some cell or cells may have their corner node or nodes (acting as intermediate DN(s)) on existing path. In such a case, DN closest to SDN is selected as new IDN for that sink and link to old upstream node towards old IDN is removed. In case sink moves to a cell none of whose corners DNs falls on existing path, a corner DN of that cell which is geographically closest to old IDN is selected as new IDN.

Once selected as new IDN, a further check is made to see whether this new IDN has only old IDN at one hop distance or does it also have any other DN on existing path including old FDN at one hop distance (FDN is the first DN on data/query path after IDN towards SDN). New IDN learns this neighbor node information from sink as sink retains coordinates of its previous IDN and previous FDN. If old IDN is the only node at one hop distance apart from new IDN, old IDN is made as FDN for new IDN and path completed. In case new IDN has old IDN as well as some other DN on existing path (may be old FDN) at one hop distance, it makes this DN as its FDN and thus modifies path at this DN by changing upstream pointer to point to new IDN and thus completes path.

Figure 5.7 shows initial scenario and shows all possible cases for sink1 movement. Arrow head in each cell indicates the selection of new IDN for sink1 in that cell. If sink1 is in cell 1 or 4, Z acts as IDN and X as FDN. If sink1 moves to 2 or 3, X is selected as new IDN and Y is FDN. For more clarity, Figure 5.8 shows new IDN selection and path setup when sink1 moves to either cell 5, 7, 9 or 11.

When other sink appears in sensor field, it first of all checks the presence of already existing grid in the field and if a valid grid is found, utilizes it. New sink broadcasts grid probe signal and its surrounding DNs hear to it. Surrounding DNs send acknowledgement to sink if valid grid is present and further initiate path setup procedure, otherwise new grid formation takes place. If grid is present, these DNs initiate geographical flooding until already existing path is intersected or SDN is reached. As per previous path setup procedure, geographical knowledge is exploited to set new path to minimum distance path. First, new sink selects its IDN as one of its surrounding DN which is closest to SDN or point of intersection of two paths. If intersection point is closer than SDN, then path from it to new IDN is set and rest of path is shared with existing one. But, if SDN is closer than intersection point, a complete new path is set. In either case, path setup is similar to setup procedure used in previous sink case. Figure 5.8 shows the path setup when sink2 and sink3 appears in the sensor field.

5.4.3 Grid Lifetime

GBDD scheme assumes that sink knows in advance the approximate period of observation, which apart from other factors hugely depends on the nature of event to be monitored. Since it is sink that triggers grid construction, therefore grid setup message includes grid lifetime value in it. Grid setup message when traverses from DN to DN provides sufficient knowledge for them regarding the time span during which present grid

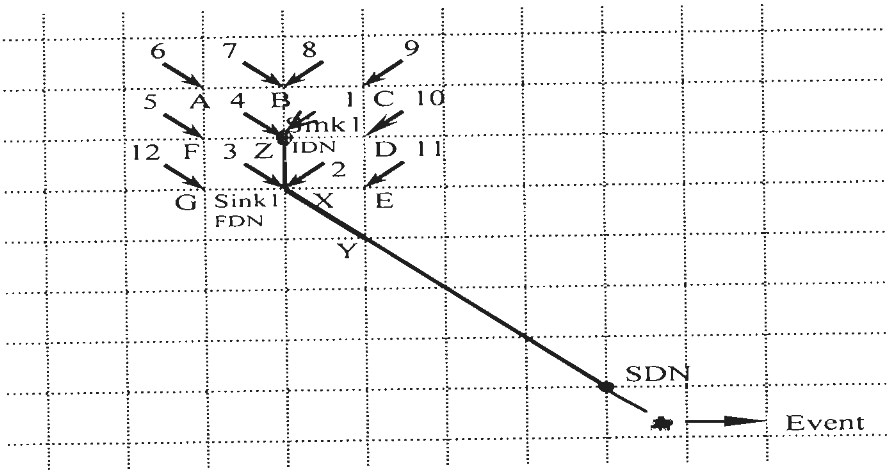


Figure 5.7. All possible cases for sink1 movement

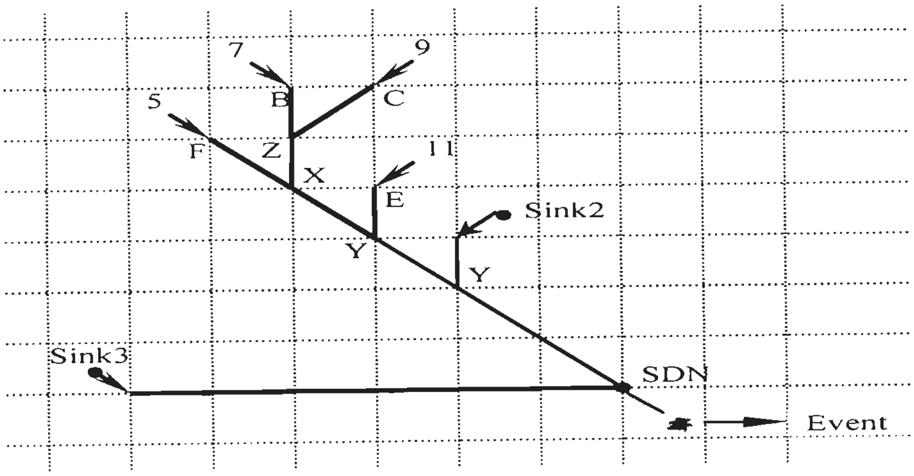


Figure 5.8. New IDN selection and path setup when sink1 moves to either cell 5, 7, 9 or 11

is valid and which is kept alive for sufficiently long period so that multiple sinks can share it without the need for new grid construction.

5.5 Performance Evaluation

In this section we evaluate the performance of proposed GBDD approach. We first define simulation parameters and performance metric. We then see the effect of various factors like number of sinks, number of sources and sink movement on the performance of GDDD and compare it with TTDD.

5.5.1 Simulation Parameters

We consider a flat and square sized two dimensional sensor field of size $2000 \times 2000 m^2$ in which 200 SNs are randomly generated and deployed. All nodes are homogeneous and each one has onboard dual mode radio i.e. high power radio and low power radio. Power parameters for high power radio are kept same as used in TTDD. This is done to compare the performance of proposed scheme with TTDD which only uses one type of radio, called high power radio here. Accordingly, values of SN's power parameters in high power mode are set as: transmitting power=0.66W, receiving power=0.395W and idling power=0.035W. However, apart from high power radio, proposed GBDD uses additional low power radio onboard. Power consumption for transmitting per bit by low power radio is much less than high power radio. For example, 802.11g consumes 112 nJ/bit as opposed to 979 nJ/bit for higher power 802.15.4. This implies that low power radio 802.11g consumes approximately 10 times less energy than high power radio 802.15.4 [29]. For simplicity of analysis, we take this approximation to scale down power consumption by high power radio by a factor of 10 to set power parameters for low power radio onboard on each SN. Accordingly, power parameters for low power radio are set as: transmitting power=0.066W, receiving power=0.0395W and idling power=0.0035W.

Table 5.2. Simulation Parameters

Parameter/ Entity	**Value**
Type of Sensor Field	Flat in two dimensional plane
Size of sensor field	2000 x 2000 m^2
Total number of SNs	200
Types of SNs	Homogeneous with dual radio
SN mobility	Stationary
High power radio transmission range	100*m*
Low power radio transmission range	25*m*
Diagonal of square sized cell of grid (*d*)	75*m* (α=75/√2m)
Transmitting power of primary radio of a node (high power mode)	0.66*W*
Receiving power of primary radio of a node (high power mode)	0.395*W*
Idling power of primary radio of a node (high power mode)	0.035*W*
Transmitting power of secondary radio of a node (low power mode)	0.066*W*
Receiving power of secondary radio of a node (low power mode)	0.0395*W*
Idling power of secondary radio of a node (low power mode)	0.0035*W*
MAC protocol	802.11
Query message size	36 *bytes*
Data packet size	64 *bytes*
Simulation period	200 *seconds*

Node is assumed capable to transmit up to 100m (R_H) while in high power radio mode and upto 25m (R_L) in low power radio mode. Therefore, according to grid construction mechanism in GBDD, the diagonal of a square sized cell is set to 75m. SN changes its transmission power accordingly while switching between high power and low power radio mode.

Links between nodes are considered bidirectional and symmetric i.e. if node A can hear from node B, then node B is also expected to hear from node A. All nodes have same transmission range(s) and there is a link between two nodes if the distance between them is less or equal to high power radio range R_H. This attributes to the fact that low power radio is used only for network management activities like formation of zone ADZ, receiving and maintaining tuples comprising information about residue energy in nodes within ADZ etc, whereas it is high power radio which is responsible for communicating queries and data among nodes. Each query packet used is 36 bytes and each data packet has 64 bytes.

Each node knows its geographical coordinates using low-cost, low-power GPS or other localization algorithms. For simulation purpose coordinates (x,y) with in the specified boundaries of sensor field are randomly generated and assigned to nodes during deployment. Each simulation run lasts for 200 seconds. Table 5.2 summarizes all simulation parameters.

5.5.2 Simulation Scenario

Figure 5.9 shows basic example scenario simulated with varying numbers of sources (i.e. events and SDNs thereof) and sinks. For clarity of diagram SNs are not shown and DNs are assumed at crossing points of grid. When first sink (i.e. sink 1) appears in sensor field and is interested in data about event 1, it triggers grid construction at A and forms virtual grid over entire sensor field. Once path is set up between sink1 and SDN 1 (i.e. SDN for first event) as per procedure given in Section 5.3.7, query and data flows DN to DN in opposite direction to each other on data/query path. Various scenarios surface with the passage of time as shown in Figure 5.9 and explained as follows:

i. Sink may slowly move away from initial location (i.e. starting point of first grid construction) and for simulation purpose maximum speed with which sinks can move is assumed to be 10m/s. In this case, it appoints nearest node to its initial location as its immediate dissemination node (IDN) and communicates with it in single hop for issuing queries or receiving data. For example in Figure 5.9, sink1

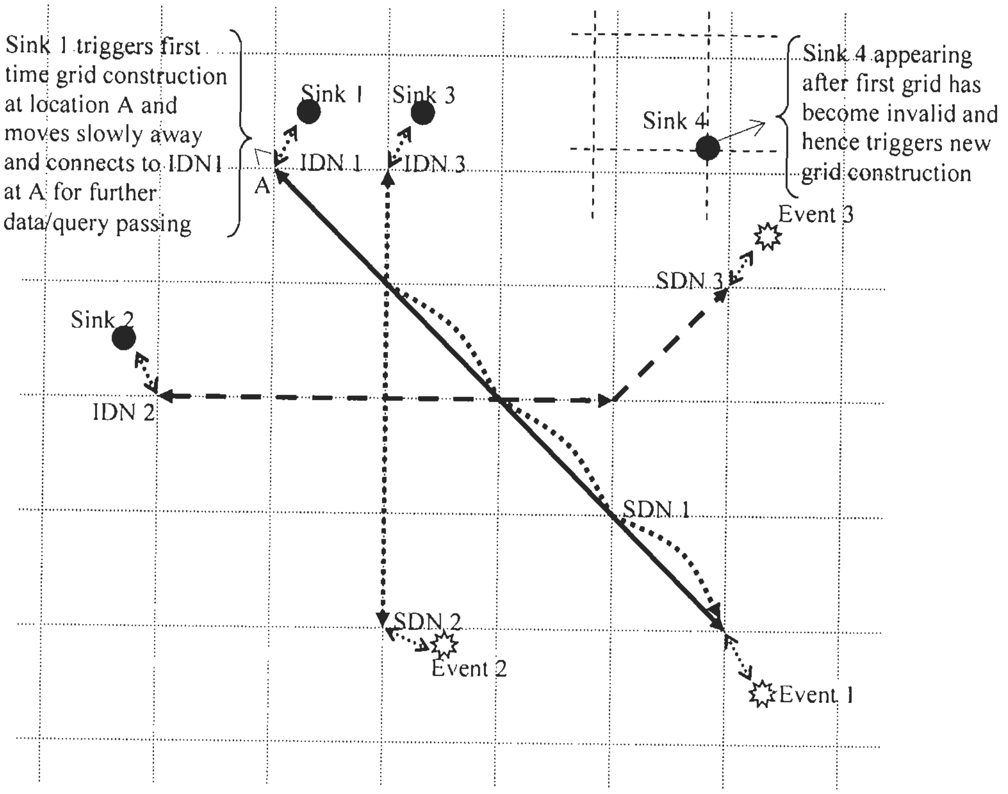


Figure 5.9. Simulation Scenario with multiple sinks and events

moves away from A and connects to IDN1. If it moves to different cell, then as per procedure given in Section 5.4, it appoints new IDN.

ii. Multiple sinks may appear in sensor field. When a new sink appears and there exist a valid grid, it utilizes it and refrains from constructing new one. For example in Figure 5.9, sink2 and sink3 shares the grid constructed by sink1. In such cases, new IDNs are appointed and paths are set between each new sink and SDN corresponding to event from which it requires data. This may lead to partially sharing already existing path between some other sink-SDN pair. However, if valid grid is not found when a new sink appears, it triggers new grid construction. In Figure 5.9, sink 4 triggers new grid construction.

iii. With time multiple events may appear in sensor field and can slowly move to any direction. For simulation purpose they are assumed to move randomly to any direction with maximum speed of 10m/s. These multiple and mobile events are handled as per procedure in Section 5.4. For example, in Figure 5.9, as new event 2 and event 3 appears in sensor field after some time, new SDNs are selected by active nodes (sensing these events) by coordinating among themselves. Also, paths towards sink (which formed grid) are created which may partially share existing path(s) between other sink-event pairs. For example, when sink 3 appears and becomes interested in data from event 1, it partially shares path already set between sink1 and event 1.

5.5.3 Performance Metrics

The main intention behind proposing GBDD is to achieve energy savings along with making it tailor made for further proposed cooperative caching. Also, the grid structure and dissemination method should not introduce unnecessary delay in forwarding query and/or data such that it becomes stale for application on reaching at sink. Hence, following two performance metrics are used to evaluate performance of proposed grid construction and data dissemination scheme when no in-network processing is performed and no caching is used.

Overall Energy Consumption

Overall energy consumption by the network includes energy consumed by all nodes in transmitting and receiving queries and data. This includes energy consumed by DNs on data/query path from SDN to IDN for a particular sink plus energy consumed by active nodes in sensing and transmitting readings to SDN added across all sink-source pairs. Like TTDD, energy consumed by nodes while in idle state is not however included as it does not reflect energy consumed in data packets retrieval.

Average Packet Delay

Average packet delay is defined as the average time between the moments a SDN transmits a packet and the moment a sink receives the packet, averaged across all source–sink pairs. Actually this metric represents average packet delay for packets emanating from all active nodes towards destined sink or sinks in response to a query or queries from that sink.

5.5.4 Effect of Number of Sources and Sinks on Overall Energy Savings

Figure 5.10(a) and Figure 5.10(b) shows overall energy consumption when GBDD and TTDD are respectively used. For comparison, Figure 5.10(c) combines both pf these results in the form of bar graph. Initially only one event (thereof only one SDN corresponding to that event) is generated in sensor field and number of sinks interested in data about that event are incrementally varied from 1 to 8 in steps of 2. With each different number of sinks, the set up is run for entire simulation period and overall energy consumed by the network is computed. Simulation is then repeated in similar manner each

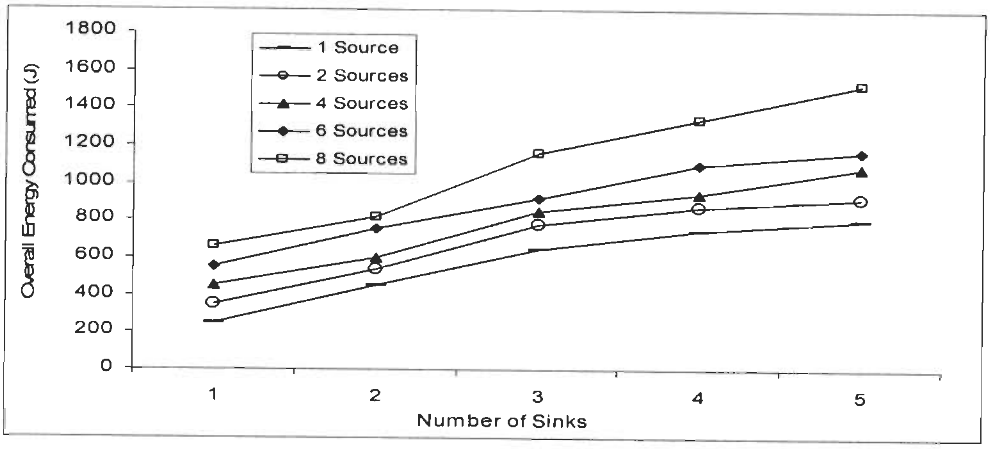


Figure 5.10(a). Effect of numbers of sinks on overall energy consumption with varying numbers of sources using GBDD

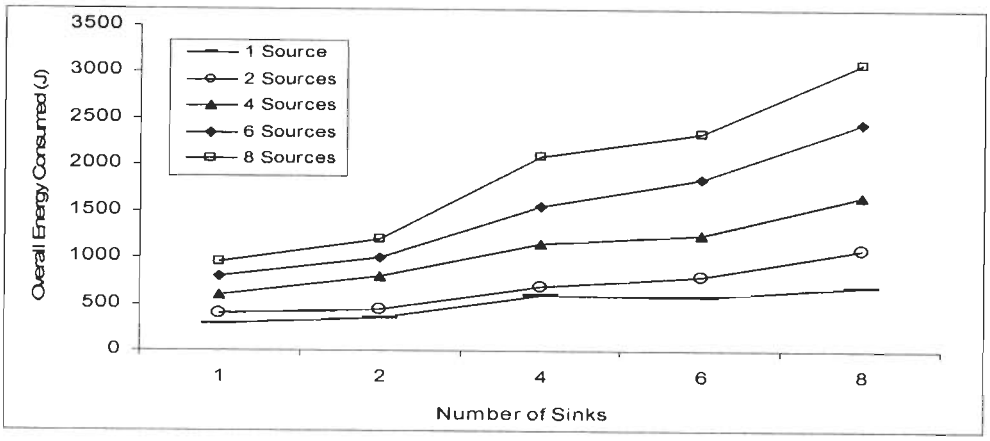


Figure 5.10(b). Effect of number of sinks on overall energy consumption with varying number of sources using TTDD

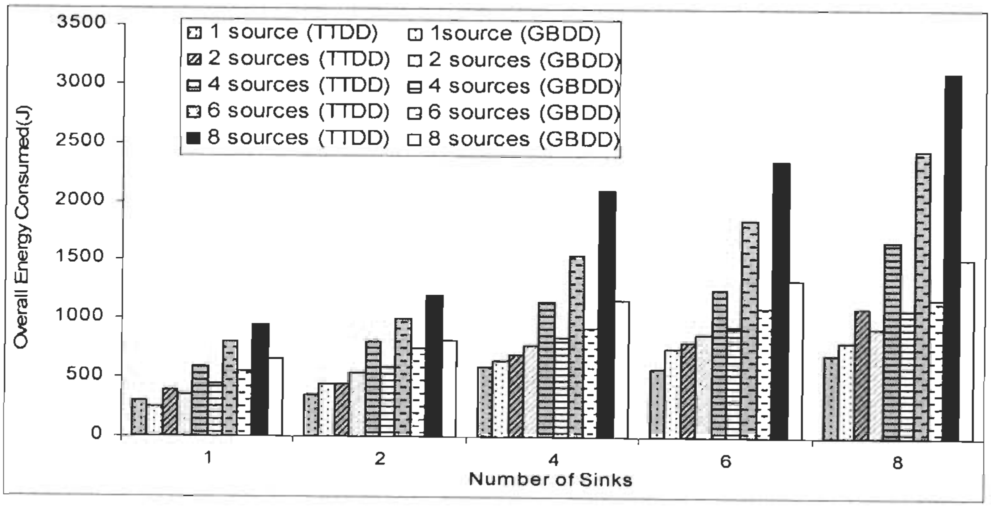


Figure 5.10(c). Comparison of overall energy consumed using GBDD and TTDD with varying sources and sinks

time with different number of sources varied from 1 to 8 in steps of 2 and overall energy is computed after each run. Sinks and events are allowed to move randomly with a maximum speed of 10 m/s. These all simulation runs are repeated for both proposed GBDD and TTDD. Energy consumptions in both schemes are shown in separate graphs to avoid cluttering.

In each approach, overall energy consumption by network increases as number of sinks increase for a given number of sources. Also, as evident from these graphs, overall energy consumption further increases with increasing number of sources. When averaged across all source-sink pairs in each approach, GBDD shows up to 43% overall energy savings as compared to TTDD.

5.5.5 Effect of Number of Sources and Sinks on Delay

Similar to computation of overall energy consumed by the network for all simulation runs as per Section 5.5.5, average packet delay is also computed. Figure 5.11(a) shows the impact of varying number of sources and sinks on average packet delay when GBDD is used and Figure 5.11(b) shows average packet delay when TTDD is used. For comparison, energy consumption in both cases is plotted in a single bar graph as per Figure 5.11(c). Results show that GBDD incurs smaller average packet delay as compared to TTDD. This attributes to the fact that unlike TTDD, in GBDD wherever possible a data packet follows diagonal path right from SDN towards sink if sink's position is quite away from it and not in straight line with its cell side. For example in Figure 5.9, path from sink1 to SDN1 crosses diagonally through many cells which might not be the case for TTDD.

If event is in straight or approximate straight line with SDN's cell side, then path is eventually shortest path (not diagonal) as communication between adjacent DNs is direct in single hop. For example in Figure 5.9, path between sink3 and event 3 is straight. Whereas in TTDD, communication between adjacent DNs is multi-hop and path from SDN to sink is not diagonal and hence paths are relatively longer. GBDD shows 30% improvement in average delay computed across all source-sink pairs for a data packet to reach from SDN to sink.

5.5.6 Effect of Sink Speed on Overall Energy Consumed

In many situations sinks are either in direct control of WSN user or if not in direct control can be accessed or relocated, whereas, appearance and disappearance of events are unpredictable in sensor field. For example, nodes with soldiers acting as sinks can be placed or moved strategically at certain locations in battlefield to collect useful data about enemy tanks entering in it. Therefore, we study the impact of sink speed keeping events'

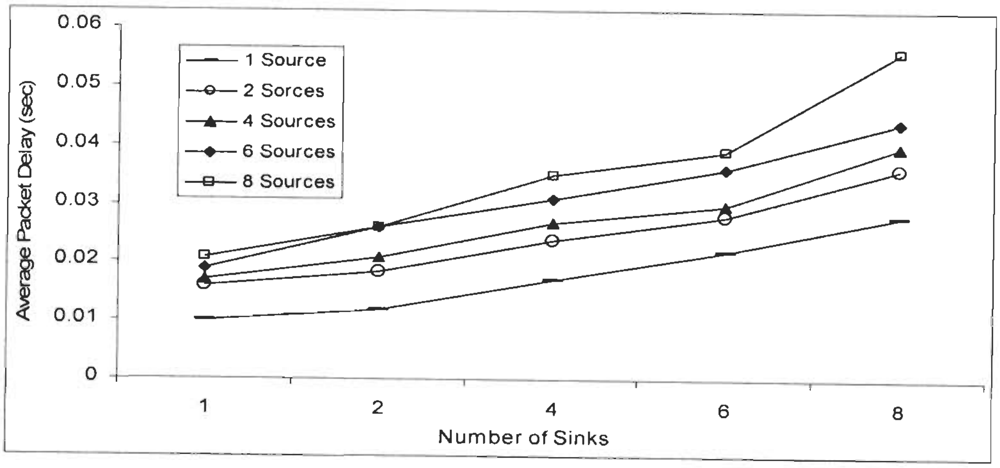


Figure 5.11(a). Effect of number of sources and sinks on average delay using GBDD

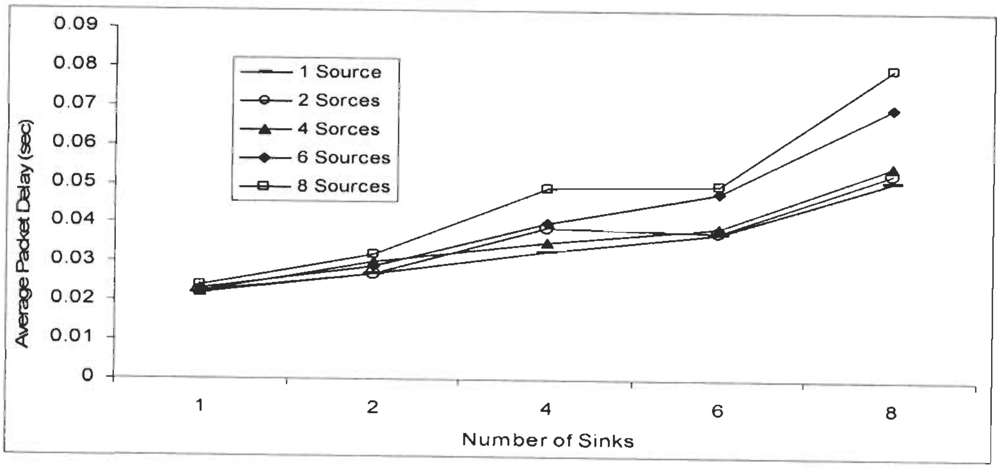


Figure 5.11(b). Effect of Number of sources and sinks on average delay using TTDD

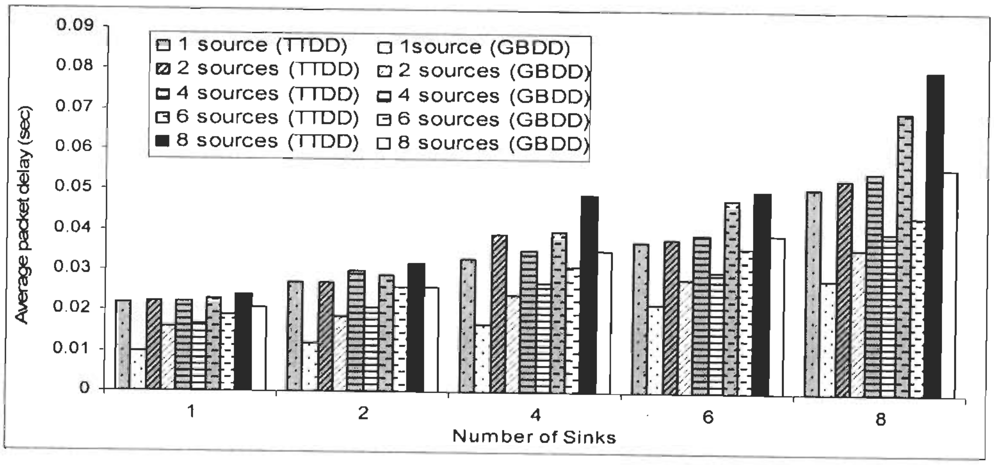


Figure 5.11(c). Comparison of average delays introduced by GBDD and TTDD

behaviour same as in earlier simulations (i.e. maximum speed 10m/s). Sink speed is varied from 1 to 20m/s. Also, in most scenarios, numbers of sinks are always relatively smaller than SDNs. Hence, we keep numbers of sinks 2 and numbers of SDNs as 8.

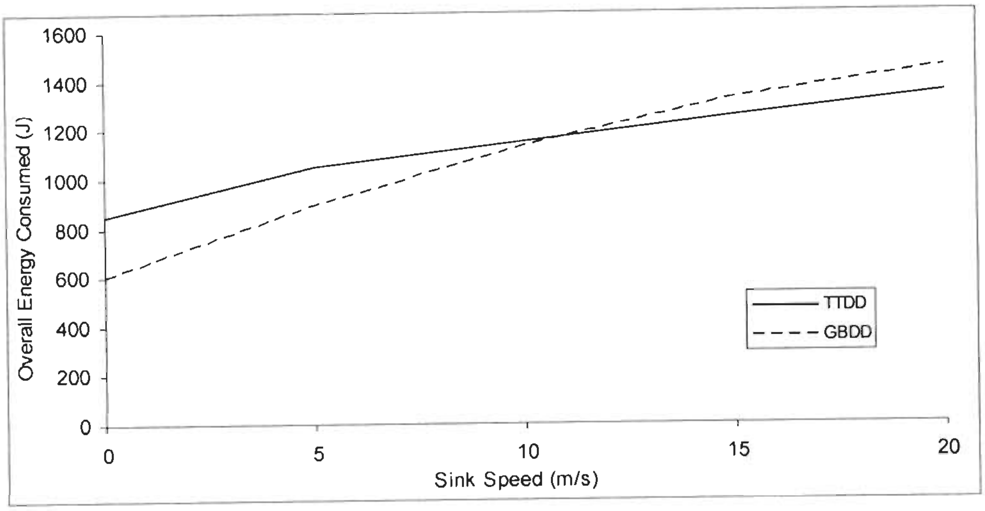


Figure 5.12(a). Overall energy consumed with varying sink speed

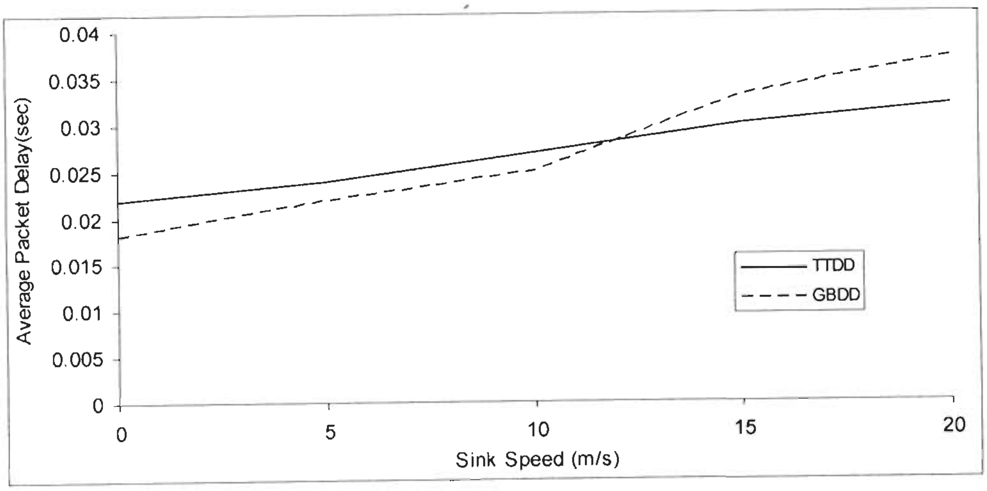


Figure 5.12(b). Average packet delay with sink speed

Figure 5.12(a) shows overall energy consumed with varying sink speed. For low sink speed GBDD consumes less energy as compared to TTDD, but when the speed is very high, GBDD however consumes more energy. This is due to the reason that as sink moves fast, it crosses boundary of a cell more frequently. Since the size of a cell in GBDD is much smaller than cell used in TTDD, cells changed per unit time in GBDD are more. Hence, more number of times sink has to find new IDN which further resolves either to link with old IDN or some other DN on path closer to it.

5.5.7 Effect of Sink Speed on Overall Average Packet Delay

With the same setup as given in previous section, average packet delays are also computed for varying sink speeds. Figure 5.12(b) shows the results when setup is run both for GBDD and TTDD separately. In case of lower sink speeds, average packet delay for GBDD is smaller than in TTDD which is again due to the same reason (i.e. shorter path shorter path) as given in previous section. However, at higher speeds due to smaller cell size in GBDD, sink crosses boundary of a cell and enters into new adjoining cell more frequently than in TTDD. It takes time to appoint new IDN for sink and to set path from this new IDN to SDN(s) from where it requires data. Path can be set by reconnecting to existing (old) path from new IDN and by sharing it or by setting new path (if sharing is not possible). This introduces delay due to path set up time and even at existing IDN packet has to wait before it is delivered to sink through new IDN. Hence, as sink moves more randomly, GBDD starts introducing slightly higher delays in packet delivery to sink and hence more average packet delay.

5.6 Conclusions

Virtual grid constructed over randomly deployed SNs has proven very useful for handling sink and event movements. GBDD exploits location awareness and dual radio mode of SNs to build grid structure over entire sensor field. Cell size of a grid is determined from the fact that all dissemination nodes communicate with their immediate neighbor dissemination nodes in single hop using high power radio. However, dissemination nodes use low power radio for other network management activities such as alternate dissemination node selection. Therefore, cell size is entirely determined using these radio ranges.

In GBDD, first sink appearing in the sensor field constructs grid with sufficiently large lifetime and is utilized by all other sinks appearing during valid period of that grid. New sink constructs new grid only when no valid grid is present. Unlike TTDD, new events appearing in the sensor field do not trigger grid construction, rather SNs sensing new event utilizes existing grid. Movement and multiplicity of sinks and events is efficiently managed through local message passing and path sharing. Simulation results reveal that GBDD gives significant improvements in overall energy savings when compared with TTDD. Also, for slow sink movements GBDD gives smaller average packet delay than TTDD. However, for higher sink speeds slightly more packet delays are introduced in GBDD than in TTDD.

Chapter 6

Cooperative Caching

6.1 Introduction

Communication is a major source of energy consumption in a WSN and hence in recent years various schemes have been proposed to reduce number of inter-node transmissions. This is achieved either by reducing amount of data to be disseminated from source to sink or by reducing path traversed by queries and data. Strategies like data compression [72], in-network data processing [36], and source data filtering [58] reduce the bulk of data packets disseminated to sink, whereas efficient routing [68], better topology management and caching reduce the length of path traversed by data and/or queries in the network.

In majority of WSN deployments, SNs continuously sense event with some sensing frequency and pump data into the network. Data dissemination method further moves data towards sink by using suitable routing protocol and/or in-network data aggregation techniques. If sink do not need this data at the time it is generated, it must be stored temporarily in the network. SNs can hold only limited data in their local storages and hence to accommodate entire data generated due to event sensing at a particular point in time, it must be spread across distributed nodes throughout the sensor field. To retrieve such information from the network when sink does not know the locations where data is spread is also a huge challenge. One of the most recent works in the area of efficient search and retrieval of information from such a WSN is by Kiran *et al.* [112] [113]. They propose couple of protocols (IRS and k-IRS) with basic principle that to route the search packet along a set of trajectories called rays, the likelihood of discovering the target information is maximized by consuming least amount of energy. The rays are organized such that if the search packet travels along all these rays, then the entire sensor area will be covered by its transmissions while minimizing the overlap of these transmissions.

Since, large number of SNs may be active (i.e. sensing event) at a given time generating sensed readings with certain frequency. This results in huge amount of data which must be disseminated towards sink. In many scenarios, sink is also one of field SNs having similar hardware constraints except possibly having radio with higher transmit/receive range. Thus, sink can easily be swamped by these large numbers of sensed readings making impossible for it to store them in its limited local storage. This results in loss of precious data which may be required by sink in future to serve queries received from base station (BS) located outside the sensor field.

BS issues queries at macro level regarding certain properties of event like 'What is the velocity of the object?' or 'What is the size of the object?' etc. In order to serve each such individual query, sink not only needs current sensor readings about event but also readings from recent past. Therefore, sink issues several micro queries towards active SNs (Figure 6.1). For example, queries for pressure and coordinates of object at times $T, T1, T2, T3, \ldots Tn$ where T is the current time and $T1, T2, .., Tn$ belongs to times from recent past i.e. $\{T1, T2, .., Tn\} \in (T-\Delta t, T)$, Δt is the size of time window. Important point to be noted here is that once a reading is used by a sink to serve a macro query from BS, it may again be required at sink to serve another macro query. This is due to the reason that reading still falls with in a time window for second query. Sink may store utilized reading in its local storage, but during its first and subsequent usage many other macro queries might have been served by sink. This causes old reading to be overwritten by some other reading due to limited storage. Hence, even if a reading has been utilized by sink and stored in its local cache, there is still a need to cache it some where in the network so that when needed, sink can fetch it again.

Even if sink is capable enough to at least receive and transmit all individual readings to base station, it soon drains its energy source. Moreover, not each individual reading is required at base station. Ideally, base station issues queries based on application (control or analysis) running there and sink serves these queries by further querying selected readings from recent past. Therefore, the issue here is to store these huge number of sensor readings generated in a given time window some where in WSN. Obvious choice seems to cache readings at nodes on data/query path from source SNs to sink, but it suffers from few problems. First, when path length between sink and active nodes (i.e. nodes sensing event) is small, the storage may still be very less to accommodate readings generated with in a given time window. Secondly, readings will be cached throughout the path from sink up to the last node towards event, where as for better performance more and more cached readings must be cached nearer to sink.

To overcome above problems, in this chapter, we propose a cooperative caching (CC) scheme which exploits cooperation among various SNs in a defined region. Apart from its own local storage a node uses storages of nodes from certain region around it to form larger cache storage known as CuMulative Cache (CMC). Proposed scheme increases data availability to sink and reduces both energy consumption and channel congestion. To increase data availability nearer to sink and to avoid unnecessary replication of a data item, a Token Based Cache Admission Control Scheme is devised where node holding token can only cache or replace data item. A proper Cache Discovery Mechanism to fetch copy of data item from a location nearest to query sender is developed. A Time-to-Live (*TTL*) based Single Copy Caching Rule is also developed to avoid sender from fetching stale data and at the same time avoiding undesired data replication. Finally, a data item utility based Replacement Policy is developed so that in case of replacement, a data item with least utility is evicted from cache.

Rest of the chapter is structured as follows. Next section describes problem scenario, network model and data/query path set up mechanism. Proposed *CC* is explained in Section 6.3 giving cache discovery method, cache admission control, single copy caching rule and item replacement policy. Section 6.4 describes results of simulation study including simulation setup. Section 6.5 concludes the work.

6.2 Problem Scenario and System Environment

6.2.1 Problem Scenario

As per Figure 6.1, BS sends a query *Q* at macro level requesting some information about the event. Examples of macro queries are:

Q1: 'What is the velocity of the event?'

Q2: 'In which direction the event is moving right now?'

Q3: 'Is the size of event fixed or changing?'

Q4: 'What is the shape of the event?'

To answer any of such macro queries sink needs multiple readings about different parameters of event taken at different points in time (i.e. current as well as in recent past) and taken by multiple SNs located near event i.e. sink exploits temporal and spatial

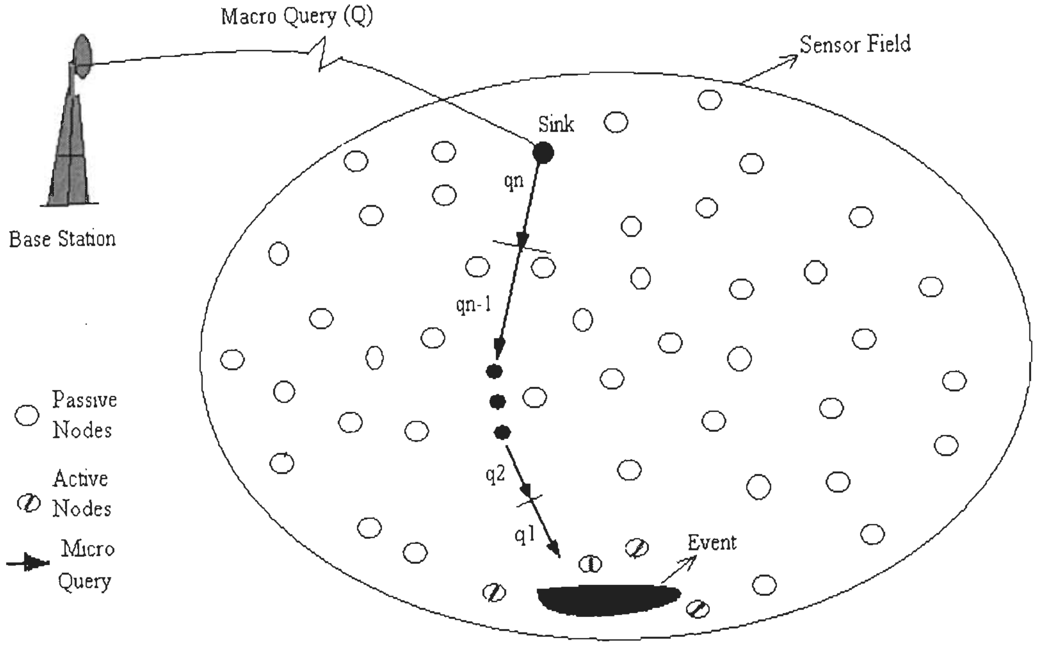


Figure 6.1. Basic Problem Scenario

correlation among sensor readings. Therefore, to serve a macro query Q sink issues several micro queries $q1, q2, .., qn$, where n is total number of micro queries. Let it be denoted as:

$$Q = q_1 + q_2 + \ldots + q_n \tag{6.1}$$

Each micro query $q_i = \langle p_j, T_k \rangle$ (for $i=1, 2, .., n$) is a request for some sensed reading about a parameter p_j (for $j=1, 2, .., m$, where m is total number of parameters) at some point in time $T_k = T - (k \,.\, \Delta t)$ where k Є [0,1]. T is the current time and Δt is the size of time window for a particular parameter type. A time window specifies a period during which all readings sampled at a SN are valid and may be utilized by micro query from sink to serve a macro query. Examples of micro queries are:

$q_1 = $ *<speed, 09:35:45>*,

$q_2 = $ *<speed, 09:35:55>*,

$q_3 = $ *<temperature, 09:36:04> etc.*

Depending upon the type, each parameter may have different window size. Size of the time window for parameters used for inferring fast changing properties of an event will have much smaller size as compared to size of time window for parameters which are used to infer rather static properties. These micro queries are forwarded towards event using single or multiple paths depending on the data/query dissemination method as used in TTDD, Directed Diffusion, LEACH etc.

If application running at base station requires some information about event more regularly than other, total possible sensing parameters in set P_T can easily be categorized in two subsets where one set (P_F) has parameters required frequently and the other one (P_I) encircles parameters required infrequently. We use a scheme where each parameter in P_F is sensed periodically and each parameter in P_I is sensed based on query received from sink. Let f be the sensing frequency at SNs having event in their sensing range, called here onwards as active nodes. During each sensing interval all active nodes sense event for every parameter in P_F. Hence, if N is the total number of active SNs and m is the number of parameters in P_F, number of readings R generated during a single time window is:

$$R = N \,.\, f \,.\, \Delta t \,.\, m \qquad (6.2)$$

R is the minimum number of readings generated during a single time window if no query requiring parameters from P_I is issued by sink during this period. Generally, all of these readings may not be utilized by micro queries generated by sink at time T (as per Equation (6.1) above) in response to macro query received from BS, still due to unpredictability of macro query they must be cached in the network. If there are *10* active nodes sensing event for 4 different parameters in P_F with f = *4 samples/second* for a time window of size of *10 seconds*, *1600* readings are generated plus few more readings in response to queries for parameters in P_I are added. The numbers of readings are generally very large to be cached on nodes on data/query path from active nodes to sink. Hence, there is a need to cache all these readings some where in the network so that when required sink can fetch these readings.

6.2.2 Network Model and Clustering

Proposed *CC* strategy can be used with any existing data dissemination scheme which either involves flat or hierarchical network topology. Once path is set up between source and sink pair, certain nodes on the path or nearby nodes may be selected to act as coordinators for forming cooperative cache zones (to be given in coming sections) to implement proposed *CC*. However, the network model and clustering scheme used in Chapter 5 for proposed GBDD is chosen here to form topology over a WSN to implement *CC*. This attributes to the fact that apart from other objectives, one of objectives of proposed GBDD scheme is to develop a data dissemination scheme involving clustering and path formation in a manner which is tailor made for proposed *CC*.

All assumptions regarding sensor field; SN having dual mode radio and location awareness; mobility of sink, event and field SN are exactly same as used in Chapter 5 for GBDD. Also, a dual radio based virtual grid is formed and maintained similarly.

However, there is a little difference in sensing scenario and data/query path set up between GBDD and proposed *CC*. In GBDD, once virtual grid is formed path setup is not initiated immediately but initiated only when presence of an event is detected by few SNs. This is due to the fact that GBDD considers pure query driven scenario where data is accumulated at SDN and subsequently forwarded towards sink only in response to a query from sink. Wherein, as evident from problem description in previous section, CC a scenario is envisioned where active SNs sense event periodically and inject sensed observations in to the network. Queries from sink may be issued at any times which traverse towards SDN(s) with the possibility of cache hits on their way. If there is no cache hit, then at SDN a special *FDSA* procedure (to be given later) is initiated to sense the event and aggregate new data item. Therefore, in CC both periodic sensing and query driven data dissemination are considered. Also, instead of a special path setup message M_{setup} used in GBDD, an initial data item is aggregated at SDN and is forwarded towards sink as explained next.

6.2.3 Data/Query Path Setup

On detecting an event first time, active SNs sense event and send readings to their SDN. To reduce the bulk of readings that need to be forwarded and stored in network, a simple data aggregation is done at each SDN. During a given sensing interval, a SDN aggregates similar readings (regarding same parameter) from each active SN in its cluster to yield a single data item. Hence, during a single sensing interval SDN generates data items equal to the number of parameters sensed during that interval and total number of readings R as in Equation (6.2) generated during a single time window are reduced to $f \,.\, \Delta t \,.\, m$ data items.

Immediately after grid formation, active nodes sense event for any one (preprogrammed) of parameters and send readings to their SDN. SDN aggregates these readings to generate initial data item $D_{initial}$ and forwards it to its upstream DN towards sink. Note that all DNs gather information about upstream neighbor during grid formation process (i.e. out of its entire one hop DNs, it selects the one which is nearest to sink). Upstream DN also forwards it to its upstream DN and so on until $D_{initial}$ reaches at sink. This sets up initial path for query and data flow from SDNs to sink. Event is assumed to be lying in one cell at a time and if it slowly moves to another cell path setup procedure

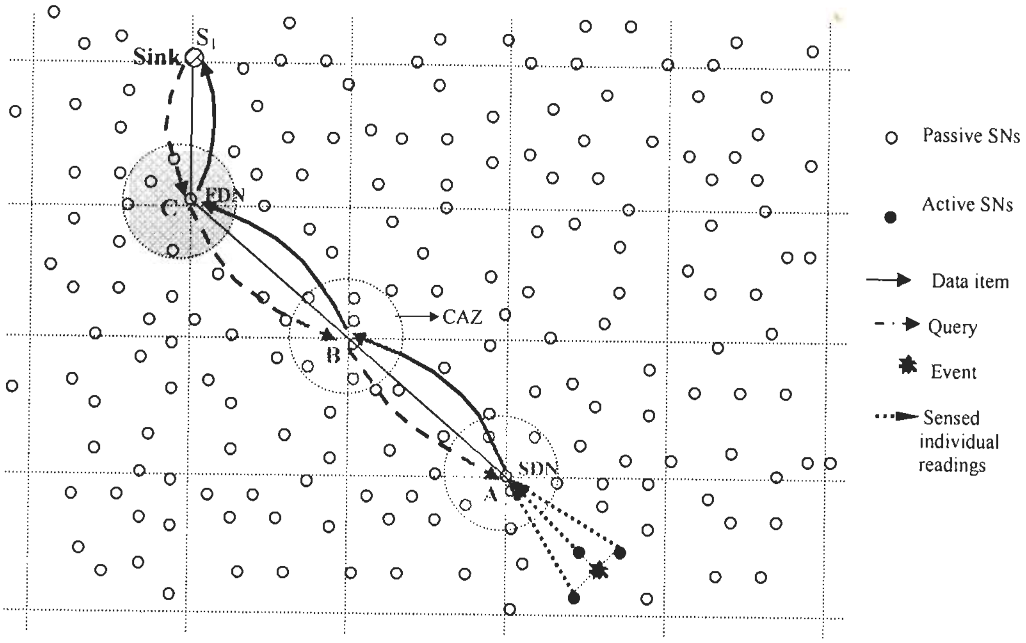


Figure 6.2. Path setup and CAZ formation

similar to GBDD takes place. Since event moves to any of adjoining cells, there is great possibility that new path will intersect existing one and this stops path setup process as it uses existing path from this point onwards to sink.

Figure 6.2 shows complete path setup scenario. After initial path set up, data items aggregated at SDN starts moving upwards towards sink and proposed token based cache admission control decides where to cache it. A query issued by sink travels DN to DN towards SDN and cache discovery procedure tries to locate it for possible cache hit. In case query reaches SDN and still there is no cache hit (possibly it requires reading about some parameter in P_l), SDN broadcasts query in its cluster using high power radio. On receiving from SDN, each active SN senses the desired parameter and sends reading to its SDN. SDN aggregates these readings to generate data item needed and forwards it towards sink to serve query.

6.3 Proposed Cooperative Caching

Table 6.1 summarizes symbols and notations used in previous sections and also in sections to follow.

Low power radio mode of a SN is exploited to form a cooperative region. In our approach, as shown in Figure 6.2, a circular region of radius $R_l/2$ around each DN including sink and SDN is defined as Cache Zone (CAZ). Let S_{CS} is the set of nodes in CAZ. Nodes in S_{CS} cooperate among themselves and with DN by sharing their local caches to realize much larger cumulative cache (CMC). All nodes in S_{CS} communicate with each other and with DN using low power radio for every cache management activity. The reason for taking radius of CAZ as $R_I/2$ is two-fold; first all cache nodes in CAZ should communicate in single hop with each other and second to utilize low power radio for caching so as to conserve energy. Each DN probes its CAZ by broadcasting special signal in it and each SN in S_{CS} responds with a tuple comprising its id (geographical coordinates) and residual energy. This way, DN becomes aware of nodes available for caching and prepares an indexed list of node-ids in descending order of residue energies. CAZ formation and path setup mechanisms yield an appropriate model for data dissemination using *CC*.

6.3.1 Cache Discovery

Before sending query for a data item, sink first of all checks its own local cache. In case data item is found there and is valid, query is immediately serviced without the need to send it any further. If data item is not found in local cache, sink looks for it in CMC

Table 6.1 Symbols and notations used

Symbol	Definition
CAZ	Caching Zone around a DN where nodes cooperate to form cumulative cache
S_{CS}	Set of nodes in CAZ
P_T	Set of all possible sensing parameters about event
P_F	Set of sensing parameters required frequently
P_I	Set of sensing parameters required least frequently
f	Sensing frequency for parameters in P_F
Δt	Time window interval during which a data aggregated at SDN may be required at sink
m	Number of elements in P_F
$D_{initial}$	Initial data item aggregated at SDN for path setup
CMC	Cumulative Cache formed by cooperating nodes in S_{CS}
FDSA	Fresh Data Sensing and Aggregation process
ACM	Active Cache Manager is a node which presently can cache and replace data items
CT	Cache Token, a special message that empowers a node to act as ACM
CIT	Cache Intention Token, a special message generated by a node showing its interest to act as ACM
TTL	Time-to-Live value for a data item
S_{min}	Minimum number of free locations to make a node candidate of ACM
A_i	Access probability of a data item d_i
a_i	Mean access rate to item d_i

formed around it by broadcasting query in its CAZ using low power radio. Node in CAZ having copy of data item in its local storage responds by transmitting data item to sink and query is serviced. In case of both local and CMC miss, sink sends query to its first neighboring DN on query/data path towards SDN. At each DN it first checks local cache for data item and then CMC formed around it. If copy of data item is not found in its local cache or CMC, query is forwarded to next DN on query/data path towards sink. In this way, query travels downstream towards SDN until data item is found at local cache or CMC of a DN.

In case copy of data item is not found in local cache or CMC of any of DNs including SDN, SDN initiates Fresh Data Sensing and Aggregation process (FDSA) to answer the query. In *FDSA*, SDN uses high power radio to broadcast a query which specifies type of parameters to be sensed. Active nodes immediately sense the required parameter about event and send it to SDN. SDN aggregates these individual readings from multiple sensors and infers the data item. Once data item is available, query is immediately serviced by sending data item towards sink following the path followed by query in reverse.

Therefore, for each data query following cases hold:

Case 1 (Local Hit): When copy of requested data item is found at sink's local cache or in CMC around it. If item is valid, it is retrieved to serve the query.

Case 2 (Local Miss): When copy of requested data item is not found in the local cache of sink or CMC around it. Sink sends query to neighboring DN on query/data path towards SDN.

Case 3 Intermediate Local Hit (I-Local Hit): When copy of the data item queried is found in local cache of an intermediate DN or SDN. Query stops traversing further and data item is sent back to upstream DN towards sink using high power radio.

Case 4 Intermediate Local Miss (I-Local Miss): When copy of the queried data item is not found in local cache of an intermediate DN including SDN. Using low power radio, CMC of that DN is queried to find the data item by broadcasting the query in CAZ.

Case 5 Intermediate Zone Hit (I-Zone Hit): When copy of the queried data item is found in CMC of an intermediate DN including SDN. Using low power radio, cache node possessing data item sends it to its coordinator DN and coordinator DN using high power radio sends it towards sink through upstream DN.

Case 6 Intermediate Zone Miss (I-Zone Miss): When copy of data item is not found in CMC of an intermediate DN excluding SDN. DN forwards query to next DN towards SDN.

Case 7 (Global Miss): When data item is not found in CMC of SDN. This is the situation where data item is not available in cache of any of DN including SDN. On Global Miss, a FDSA process is initiated.

While data item moves upwards towards sink one hop at a time (i.e. DN to DN), a Cache Admission Control strategy at each DN decides whether to cache data item at its local cache or in CMC or not to cache at all. Further, in case a data item is need to be evicted out of local cache or CMC at any node, a proper Item Replacement Policy is devised to accommodate new item in case cache is full.

6.3.2 Token Based Cache Admission Control

As explained in Section 6.2.1, data items pertaining to parameters in set P_F are periodically generated at SDN and forwarded towards sink whether all of them are immediately required or not. Also, when query is issued by sink, it travels towards SDN and is served either by cache hit (local or CMC) at some DN or by initiating FDSA procedure at SDN (if it require data which is not cached in the network). In both above cases data items travel towards sink. Also, as per Section 6.1, a data item once utilized by sink to serve a macro query may again be required to serve another macro query since it still falls in a valid time window of second macro query. As sink's local storage is also limited, during first and subsequent usage of a data item many other macro queries might have been served by sink. This causes old data item to be overwritten by some other data item. Hence, even if a data item has been utilized by sink and stored in its local cache, still there is a need to cache it elsewhere in the network so that when needed, sink can fetch it again from the nearest location.

To increase proximity of the data items nearer to sink, it is always better to start caching data items nearer to sink. First of all data items are cached at sink (in its local cache and CMC), then at its downward (towards SDN) DN's cache (local cache and its CMC), so on until SDN's local cache and its CMC is full. But, this suffers from following problems:

i. When a data item reaches at a DN or is first time aggregated at SDN, how this node knows whether to cache data item or forward it towards upstream DN? If it forwards data item without caching and there is no free space at any of upstream DN, data item might have to be transmitted back downwards for caching causing major overheads or might have to be accommodated by evicting some valid data item at some upstream DN which itself may be required soon. On the other side, if on the availability of free space a node caches data item before forwarding, multiple copies of a data items may exist and except for one none of which is utilized ever.

ii. To solve above problem, there seems a simple approach that some flag is set or some other indication is provided at each DN to show whether its cache (local and CMC) is full or still has space to accommodate a data item. Once cache is full, node can pass this indication to downstream DN enabling downstream DN to decide on data item crossing through it. In this way caching can start from sink towards SDN. But, as each data item has a Time-to-Live (*TTL*) value and on its expiry data item becomes stale. While a downstream DN is caching, many data items at an upstream DN may become stale and thus provides opportunity to cache data items at this upstream DN which is nearer to sink. How can a downstream DN know about such an upstream DN?

To solve above problems, a token based cache admission control scheme is devised and is given as in flowchart in Figure 6.3. In this scheme, no DN other than the one possessing a token known as cache-token (*CT*) caches data item. *CT* is a simple small sized message which tells a DN to be ready for caching data items crossing through it and all other DNs simply forward data items. *CT* passing between DNs is regulated both by sequential flow and another special token called as cache-intention-token (*CIT*). *CIT* is used by a DN to show its intention to cache data items and is explained later.

For cache admission control, there are two cases.

***Case 1:** When CT has not yet completed a circle from sink-FDN-IDN-…-IDN-SDN-sink*

Initially, sink possesses *CT* and caches all data items and no other DN caches data items at this point. A node (sink or any DN) which is currently active for caching is called here onwards as Active Cache Manager (ACM). When sink's local cache is full, it next uses CMC formed by cooperating nodes in its CAZ. Using low power radio sink broadcasts data item and id of a node where data item is to be cached so that no other than the node specified by id caches its copy. Nodes from CAZ are selected sequentially as per indexed list formed at sink as above i.e. first node with highest residue energy, then second highest and so on until all nodes in CAZ exhaust.

Once CMC is also full, sink finishes the role of ACM and passes *CT* to its first neighboring DN (called FDN here onwards) on data/query path towards SDN. This tells

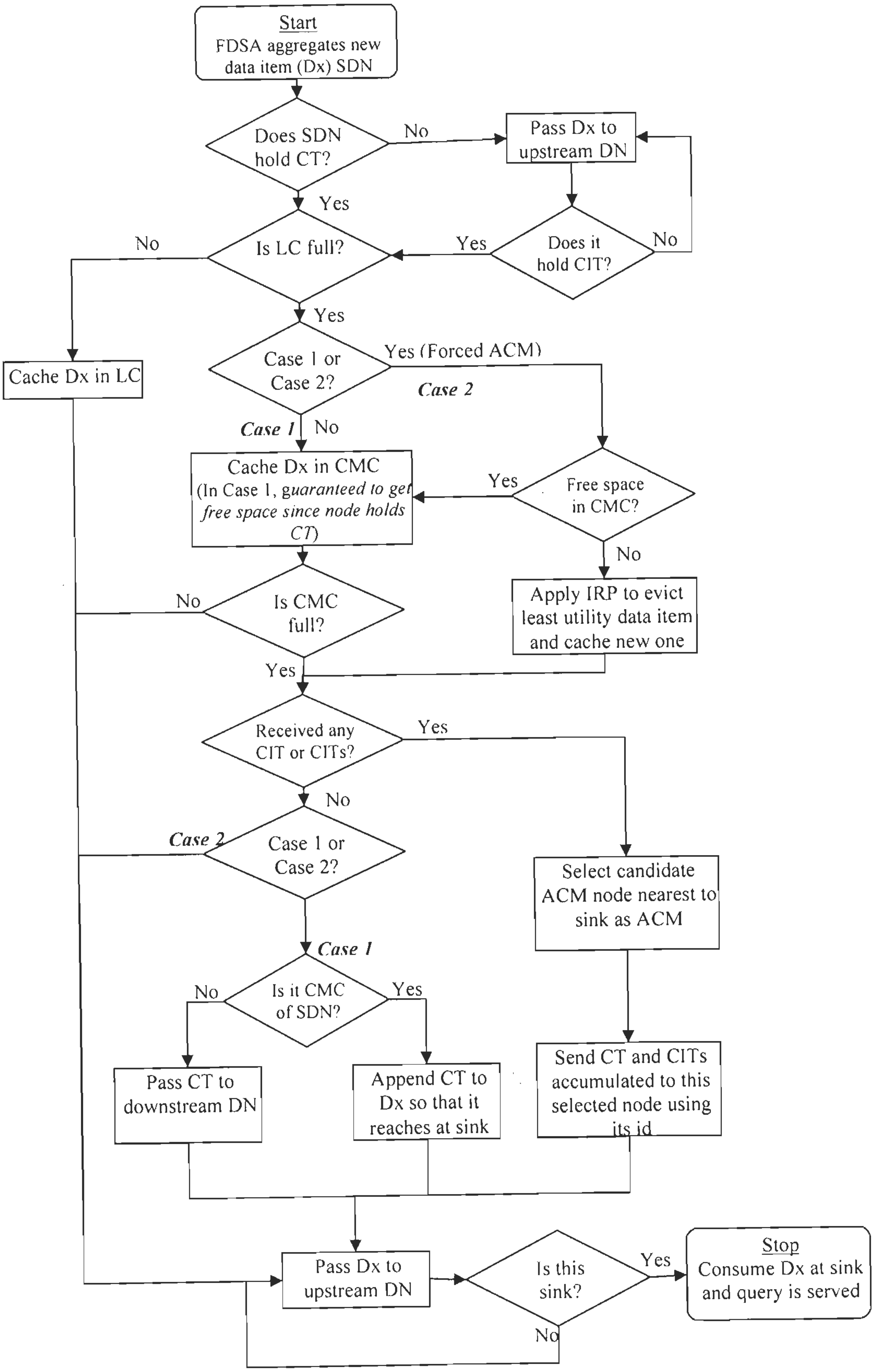


Figure 6.3. Token based cache admission control

FDN to get ready to act as ACM. A DN including sink must have some minimum number of storage locations (S_{min}) free in its cache (local and CMC) to act as ACM, otherwise *CT* is simply forwarded to next downstream DN. If FDN becomes ACM, like sink it first caches data items crossing through it in its local storage and then in its CMC. Once its CMC is exhausted, it passes *CT* to its downstream neighboring DN. This way caching and token passing continues until ACM is found or cache of SDN is exhausted. On exhausting SDN's cache *CT* is forwarded to sink with special indication for intermediate DNs to forward it upwards until it reaches at sink and circle is completed. However, sink may not be ready to act as ACM and hence item replacement policy comes in to play. Figure 6.4(a) shows such free falling *CT* passing scenario.

However, cached data item at local cache or CMC of a DN may become stale due to expiry of its *TTL* value and corresponding location thus becomes available to cache new data item. If number of available free locations in cache (local and CMC) reaches some minimum number S_{min}, DN becomes candidate to act as ACM. To increase cached data proximity nearer to sink, this DN proactively sends its intention to act as ACM by sending Cache-Intention-Token (*CIT*) to neighboring downstream DN and neighboring DN keep passing it to its downstream DN until it reaches at a node acting as ACM. When an ACM receives *CIT* from upstream DN, it passes *CT* back towards this upstream DN. On receiving *CT*, this upstream DN now becomes ACM and proximity of cached data nearer to sink is thus increased. Also, if *CIT* from multiple upstream DNs is received, the one nearest to sink is preferred. The reason for sending *CIT* only downstream is that all upstream DNs are nearer to sink than DN itself and sending *CIT* to them will rather decrease the cached data proximity. Figure 6.4(b) shows such a situation where *CT* is occasionally pulled back towards sink by *CITs* issued from some upstream DNs. While DN B is acting as ACM it receives *CIT* from DN A and hence passes *CT* back upstream to A. After exhausting its cache, A again passes *CT* back towards downstream node B. At this time if B is not candidate for ACM (i.e. S_{min} data items do not become stale), it further passes *CT* downstream towards SDN.

Therefore, *CT* normally moves in a circular fashion from sink-FDN-IDN-IDN-....-SDN-sink following *free-falling* approach, but also takes backward upstream moves towards sink on receiving *CIT* from upstream DNs. If average query generate time is large and/or *TTL* values of data items are small, there are less chances of *CT* reaching circularly back at sink. In this scenario, *CIT* keeps on pulling *CT* back towards sink. On the other hand, if queries are issued much frequently and/or *TTL* is large, there are chances that *CT* may complete the circle often. Also, if average number of SNs per CMC is less and/or

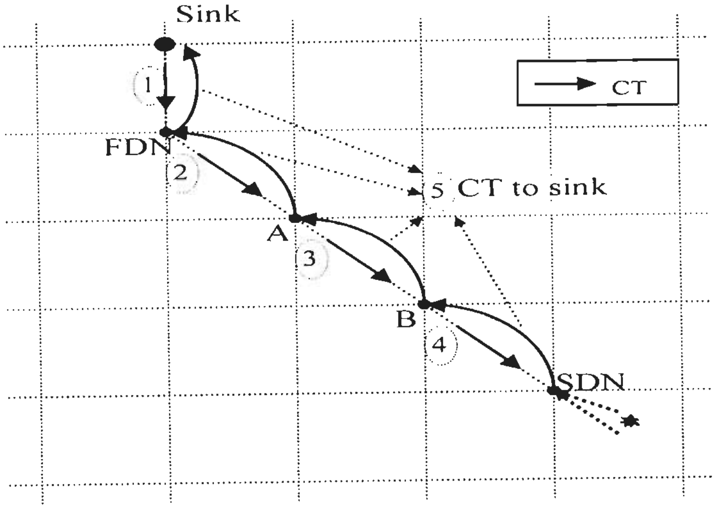


Figure 6.4 (a). Token passing without backtrack

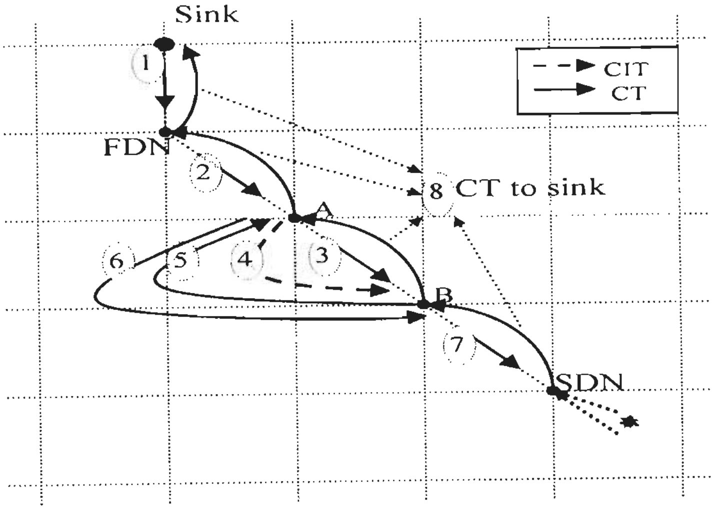


Figure 6.4 (b). Token passing with backtrack using CIT

path length from sink to SDN is small, chances of exhausting entire cache in the network are more and hence chances of completing the circle are more.

Case 2: Once CT completes circle from sink-FDN-IDN-...-IDN-SDN-sink

When SDN's CMC is exhausted, SDN sends *CT* back to sink on data/query path and completes circle first time. Since sink may not be at a single hop distance from SDN, *CT* moves hop by hop towards sink and each DN remembers the direction of *CT*. Once this circle is complete, the free-falling nature of token passing as in case 1 diminishes. This is due to the fact that if no downstream DN has S_{min} locations free to become ACM, the free-falling nature may continuously force *CT* to move around circle without finding a candidate for ACM.

Therefore, when *CT* after completing circle reaches at sink and sink even does not have at least S_{min} locations available (i.e. either completely full or $<S_{min}$ available locations), still it is forced to act as ACM (called forced ACM). If all locations at sink are occupied, it uses item replacement policy to overwrite some existing data items to cache new ones and if some locations ($<S_{min}$) are available, it first utilizes those before applying item replacement policy. Unlike case 1 where next downward DN is assured candidate for ACM, here downward DN may not have S_{min} locations available. Hence, present ACM (here sink) does not pass *CT* to next DN and keeps on acting as ACM. But, if item replacements are done at same node for long (until it receives *CIT* from somewhere), chances of over writing fresh data items are more which causes loss of valuable information. Therefore, a limit is put on number of replacements made at a forced ACM. One of the choice is to take equal to the number of data items in ACM's cache (local and CMC). If this limit is met, node passes *CT* to next downstream DN to make it next forced ACM. If sink (ACM here) receives *CIT* from any DN, it immediately forwards *CT* towards it to make it ACM. In case of multiple *CITs* received from DNs, sink passes *CT* to the one nearest to itself.

When a DN other than sink is forced ACM, it behaves similar to sink above. Unlike case 1 where *CITs* are considered only from some upstream DNs, it can receive *CITs* from any direction and passes *CT* only on receiving *CIT*. All DNs are aware that circle is complete and also know the direction of *CT* because *CT* from SDN to sink has passed through them. Hence, whenever any DN including SDN and sink has at least S_{min} free locations in its cache, it sends *CIT* toward current ACM irrespective of the fact whether it is sending downstream or upstream. Current ACM may receive multiple *CITs* from two or more downstream or upstream DNs and one nearest to sink is selected as next ACM. New ACM is also provided with the information of other candidate ACMs by old ACM. As shown in Figure 6.5(a), B receives *CITs* from FDN, A and SDN. It selects FDN as new ACM and provides it information about other candidate ACMs (A and SDN). Figure 6.5(b) shows *CT* being passed to FDN while intermediate nodes (here only A) simply forwards it and note the direction. This also avoids A and SDN from sending their *CITs* again.

6.3.3 Single Copy Caching Rule

Due to factors like multi hop environment, changing nature of sensing scenario, power and bandwidth constraints natural choice for cache consistency is towards weak consistency model based on *TTL* value. *TTL* value for any data item is heavily dependent

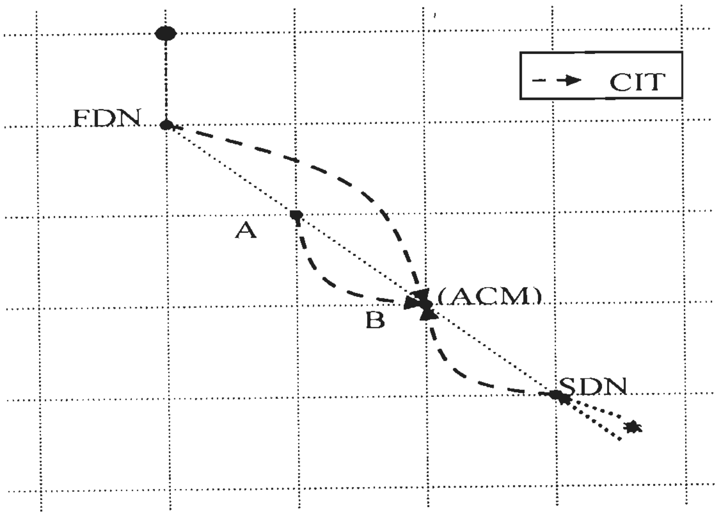


Figure 6.5(a). While acting as ACM, B receives CITs from FDN, A and SDN

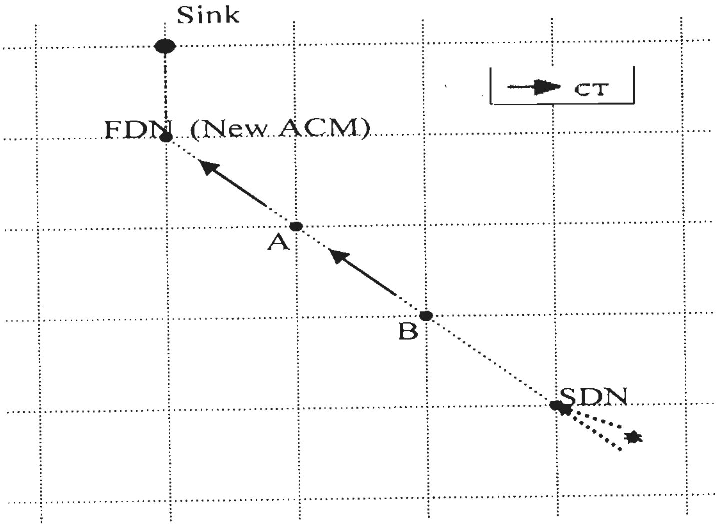


Figure 6.5(b). B selects FDN as new ACM since it is nearest to sink.

on its type, event characteristics and application requirements at sink. For example, if data item is about size of an object, its *TTL* value may be much larger than *TTL* value for item representing its location. This is due to the reason that size is relatively static, whereas it may change location frequently. Also, if application at sink requires data about an event with much higher precision both in terms of time and position, *TTL* value for a data item should be very small. *TTL* value for a data item is set when it is first time aggregated at SDN as a result of FDSA.

Data item moves from SDN towards sink in response to query from sink and its copy is cached only at ACM. Hence, if *CT* does not backtrack at all to upstream DN, only single copy of data item is cached in network. However, in practice *CT* may backtrack on receiving *CIT* from upstream DN and thus multiple copies of data item may exist. For example in Figure 6.4(b), suppose B is acting as ACM and caches a data item *Dx* crossing through it. B does not know whether *Dx* is fetched from cache of some downstream DN or is freshly aggregated item at SDN. If *Dx* is fetched from some downstream DN's cache, two copies of *Dx* are cached in the network. Further, while B is acting as ACM and receives *CIT* from any upstream DN, say A, it passes *CT* to A (to A as ACM). If sink again issues a query for *Dx*, it reaches at B to find copy of *Dx* in its cache and while this data item *Dx* moves towards sink, it passes through current ACM A which caches *Dx* in its cache. Likewise, multiple copies of data item may exist which in present scenario is total wastage of storage.

Also, as query always moves from sink downwards, the first copy of data item nearest to sink always serves the query, while other copies are never used. Therefore, a rule is used that ensures only single copy of a data item at node closest to sink or at sink itself. The rule becomes:

Rule: "*If a query is serviced by cache hit at a DN one of whose upstream DN or sink is ACM, DN immediately deletes copy of data item from its cache and frees storage.*"

If there is a cache hit for query issued by sink at a DN, copy of data item moves upstream towards sink. If one of upstream DN or sink to current DN is presently ACM, it also caches data item when it reaches there. In this way two copies of same data item are created. Subsequent queries for the same data item are always served by copy at this ACM instead of downstream DN (data item's previous location) since ACM comes first on query flow path. Since all subsequent queries also follows same path, copy of data item at downstream DN is never used and hence wastes storage. Also, as explained earlier each DN knows the direction of current ACM because it observes the direction of movement of *CT* while crossing through it.

This single copy strategy is beneficial for WSN scenario both due to limited storage capacity of SNs and relatively static positioning of sink.

6.3.4 Least Utility Based Item Replacement Policy

As explained in Cache Admission Control, there comes a situation when cache (local and CMC) of a DN (including sink and SDN) is full and new data item must be stored there. To accommodate new data item there is no option other than overwriting an existing data item in cache. The victim must be selected based on some policy which results in minimum loss of information and minimum overheads.

Active nodes sense multiple parameters from the event and hence different types of data items are aggregated at SDN. Also, depending on the type of event property measured, different data items have different values of *TTL*. Hence, data items become invalid randomly in cache despite the fact that initially they are stored sequentially one after another in the order of arrival in cache. Data items are first stored in local cache of a node and then starting with the first cache node in set S_{CS} till the last node in the set (Note that nodes in S_{CS} are sequenced in decreasing order of remaining energies). As per Chand *et al.* [97] a utility based criteria to find an item with least utility is used. But utility function is computed from two factors:

Popularity: The access probability reflects the popularity of a data item for a sink. An item with lower access probability should be chosen for replacement. Access probability A_i for data item D_i in local cache of a node is given as:

$$A_i = a_i \Big/ \sum_{k=1}^{N} a_k \tag{6.3}$$

Where a_i is the mean access rate to item D_i and N is the total number of data items in the local cache of a node. If data item is at CMC, it must be stored in local cache of one of SNs in S_{CS}. Access probability of this data item is also calculated with respect to total number of data items in local cache of that particular SN and not total number of data items in CMC. However, if access probability of a data item with respect to total number of data items in CMC is computed, a large amount of overhead is required every time to compute access probability. This overhead includes communication among all nodes forming CMC and may prove costlier in terms of energy consumption and delay. Therefore, while dealing with CMC, one SN from S_{CS} at a time is taken to find victim data item for replacement. This node selection to find victim data item in CMC takes same ordering of nodes as taken while filling cache first time.

Lifetime: An item with shorter *TTL* value remains valid only for limited period and hence is better candidate for replacement. But, if its access probability is high, it may still be accessed many times in its limited remaining valid time.

Based on above factors, utility of data item D_i is computed as follows:

$$utility_i = A_i \,.\, TTL_i \tag{6.4}$$

Item replacement policy based on utility of a data item is proposed as follows:

i. *Data item is replaced only from the cache (local or CMC) of the ACM.*

ii. *Data item with least utility value as per function (6.4) is first replaced in local cache of the DN. This continues until N new data items are replaced in local cache, where N is the maximum number of data items that can be stored in local cache of a node. The reason for replacing only N data items is to avoid repeatedly overwriting recently overwritten data items which have comparatively larger TTL values than data items at some other SN forming CMC.*

iii. *If N replacements have already been made in local cache of a DN holding CT (i.e. ACM) and new data item is to be cached, it is cached in CMC (i.e. nodes in S_{CS}) of that DN. Nodes in S_{CS} are also selected in the same order they are selected during cache admission and victim data items are found. In each node N replacements are done.*

iv. *Immediately after N^{th} replacement in last node of S_{CS}, CT is passed to downstream DN making it forced ACM. Any new data items are cached in new ACM's cache, either at some available locations or by applying above item replacement policy.*

A complete overview of all cache management modules is shown in Figure 6.6.

6.4 Performance Evaluation

In this section performance of proposed *CC* for WSN is evaluated using simulation.

6.4.1 Simulation Model

A flat sensor field is assumed flat of size 700x700m^2 where 500 nodes are randomly deployed. High power transmission range (R_H) of a SN is taken as 100meters and low power transmission range (R_L) as 25meters. As per GBDD grid construction mechanism, first appearance of event in the sensor field forms a grid of equal square sized cells of diagonal 75meters (R_H-R_L). All nodes whether normal SNs, DNs or sink are considered stationary.

BS issues query at macro level about some property of event and sink converts macro query into a number of micro queries. For simulation purpose, it is assumed that each macro query on an average results in to four micro queries from sink. Each micro query needs a valid data item aggregated at SDN. Performance evaluation here is in reference to these micro queries. Sink issues micro query for required data by forwarding it towards SDN on data/query path and data item flows in reverse on same path towards sink. Sensing frequency is always set such that required data item about a parameter from

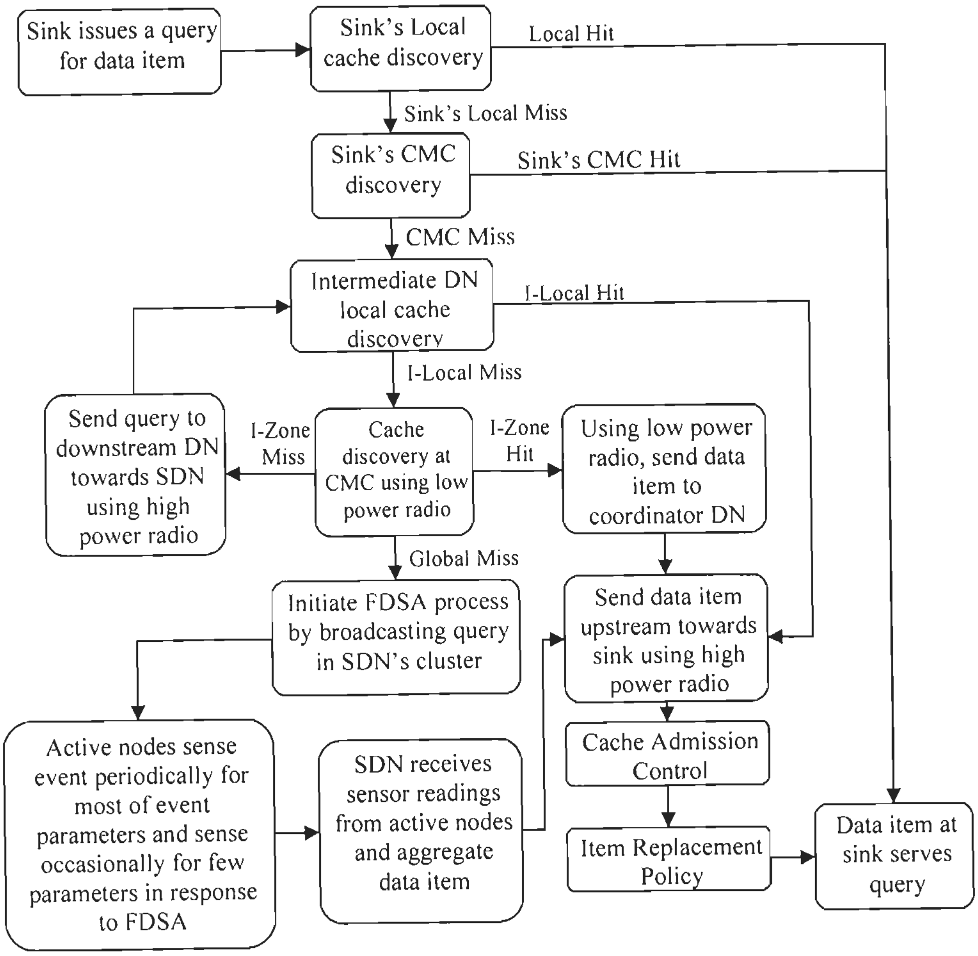


Figure 6.6. Overview of complete query generation and its service scenario

P_F with in a time window is always aggregated at SDN prior to its demand. Only, very few data items about parameters from P_I are aggregated on demand at SDN. Still data item may not be available to serve a query due to various reasons such that lack of storage space and replacement by some other data item etc. Therefore, entire issue is between efficient cache management (cache admission, cache discovery, item replacement etc.) and limited storage of a SN.

In first set up, Poisson distribution is used to generate micro queries with mean query generate time of 0.2s to observe the effect of cache size on evaluation metrics. In second set up, mean query generate time is varied from 0.1s to 1.2s while keeping cache size fixed. We categorize all data items under 10 classes, where data item from each class has different Time-to-Live (*TTL*) value. Random numbers are generated to represent data items following normal distribution over all classes and these numbers are generated with a gap following Poisson distribution to simulate query generate time.

All SNs manage small portion of their memory as cache where they can store data items and are able to delete them when their *TTL* values expire. Table 6.2 gives summary of parameters used for simulation and are varied to study their effects.

6.4.2 Performance Metrics

Metrics used for performance evaluation are *number of cache hits, number of item replacements and energy consumed* by the network.

Number of cache hits

Number of cache hits gives the total micro queries served from cache (local or cumulative) without the need to fetch data by initiating FDSA process. This ultimately measures the effectiveness of the proposed *CC* scheme to serve macro queries. Cache hits are further shown as hits at DNs' local caches (local hits) or at CMC formed (CMC hits).

Table 6.2. Simulation Parameters

Parameter	**Value**
Size of sensor field	700 x 700m^2
Total number of nodes in sensor field	500
High power transmission range of a node (R_H)	100 meters
Low power transmission range of a node (R_L)	25 *meters*
Cell size	Square with 75*m* diagonal
Data item and query size	64 *bytes*
Default cache size (however varied in some experiments to observe its effect)	320 *bytes* (can hold 5 data items)
Initial energy level of each node	10 *joules*
Simulation time	200 *seconds*
Routing protocol	None
MAC protocol	802.11
Network type	Homogeneous
Node mobility	Stationary
Energy consumed in transmitting data or query in high power mode	0.0024 *J*
Energy consumed in transmitting data or query in low power mode	0.0010 *J*
Energy consumed in receiving data or query in high power mode	0.0018 *J*
Energy consumed in receiving data or query in low power mode	0.0008 *J*

Number of item replacements

Number of item replacements performed indicates the capability of the scheme to retain valid data items in cache. Smaller number of data item replacements indicates better data items retaining capability in cache.

Energy consumed

E*nergy consumed* by the network sums up energy consumed by all nodes involved in data retrieval on data/query path right from sensing nodes around the event up to the sink. Energy consumed by nodes other than nodes involved in data routing and caching is not taken in to consideration.

6.4.3 Results

Effects of workload parameters such as mean query generate time and cache size on percentage of cache hit ratio, number of item replacements and overall energy consumption by the network are studied. Performance of proposed CC scheme (CoCa) is compared with with non-cooperating caching scheme (NCoCa) which uses same data dissemination/routing scheme as used in proposed cooperative scheme but does not form CMC i.e. only local caches of DNs on the path are used. Same metrics are used to compare performance of CoCa with a scenario where no caching (NoCa) is used at all.

6.4.3.1 Effects of Mean Query Generate Time

Figures 6.7(a), 6.7(b), 6.7(c) and 6.7(d) show effects of mean query generate time T_{mean} when cache size is fixed at 5 data items (i.e. it can accommodate maximum 5 data items). Figure 6.7(a) shows how number of cache hits varies in cooperative cache scenario and non-cooperative cache scenario with T_{mean}.

Initially, it results in more cache hits both for CoCa and NoCoCa and reduces significantly with increasing T_{mean}. Especially in CoCa, initially the number is much higher than NoCoCa because DN in CoCa has more number of cache locations available than in NoCoCa. Due to small T_{mean} data items in cache remain valid for comparatively large number of queries issued by sink which results in more cache hits. However, when T_{mean} is large, the difference in cache hits in CoCa and NoCoCa vanishes due to the fact that data items in cache become stale due to expiry of *TTL* despite cache having sufficient free locations.

Figure 6.7(b) shows the breakup between local and CMC cache hits. The point to be noted here is that the percentage contribution of CMC hits to total hits decreases with

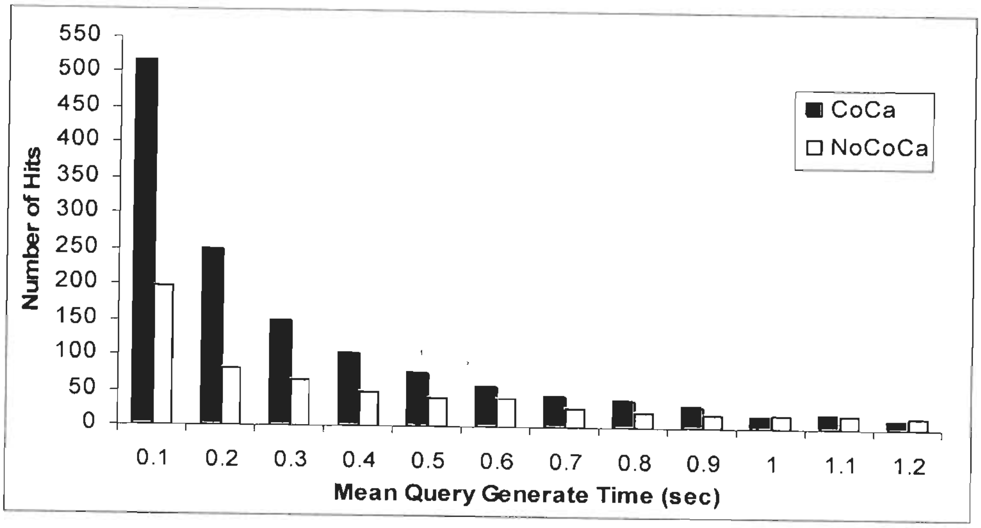


Figure 6.7(a). Effect of Mean Query Generate Time on Cache Hits in CoCa

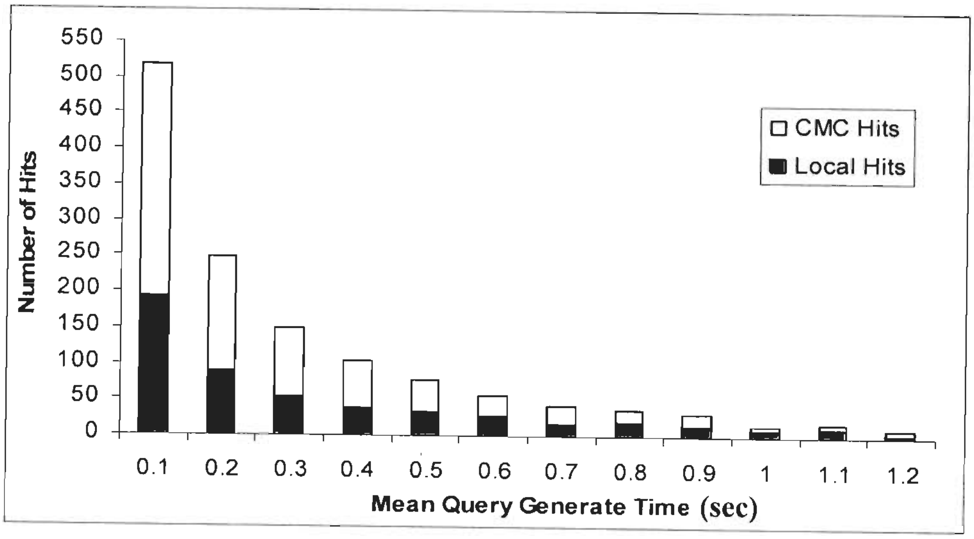


Figure 6.7(b). Effect of Mean Query Generate Time on Local and CMC Hits in CoCa

increasing T_{mean}. This is due to the reason that our token based cache admission control first fills local cache and then starts filling CMC, therefore for large T_{mean}, data items in local cache keep on becoming stale, obviating the need to push data in CMC.

Figure 6.7(c) shows the impact of T_{mean} on number of data item replacements R_p. R_p is significantly less for CoCa than for NoCoCa which attributes to large number of CMC available in CoCa. For CoCa, R_p becomes zero much earlier than in NoCoCa. This is further due to the fact that in CoCa, number of cache locations per DN is more and with increasing T_{mean} the number of data items becoming stale exceeds the number of new data items to be cached. Figure 6.7(c) clearly shows the difference in limits of T_{mean} for CoCa and NoCoCa when R_p becomes zero.

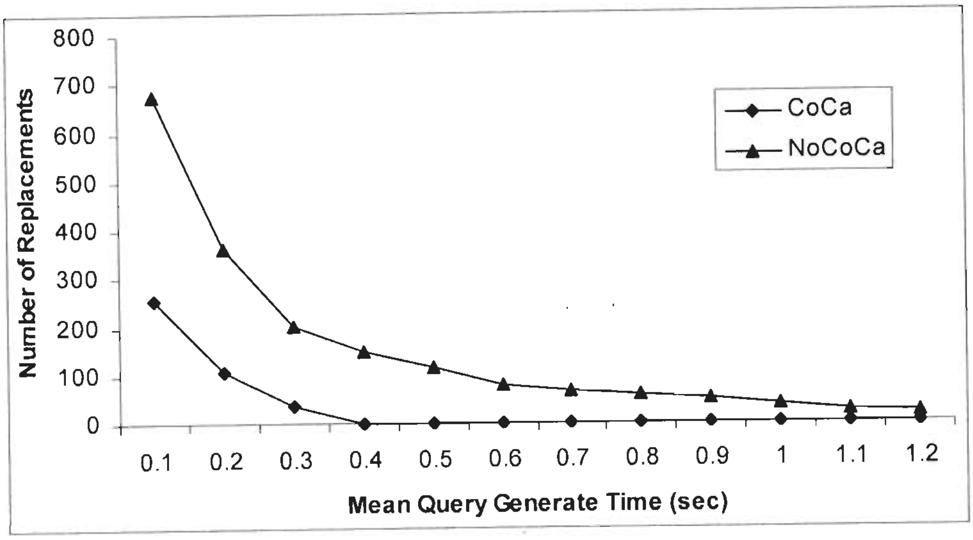


Figure 6.7(c). Effect of Mean Query Generate Time on Number of Data Item Replacements

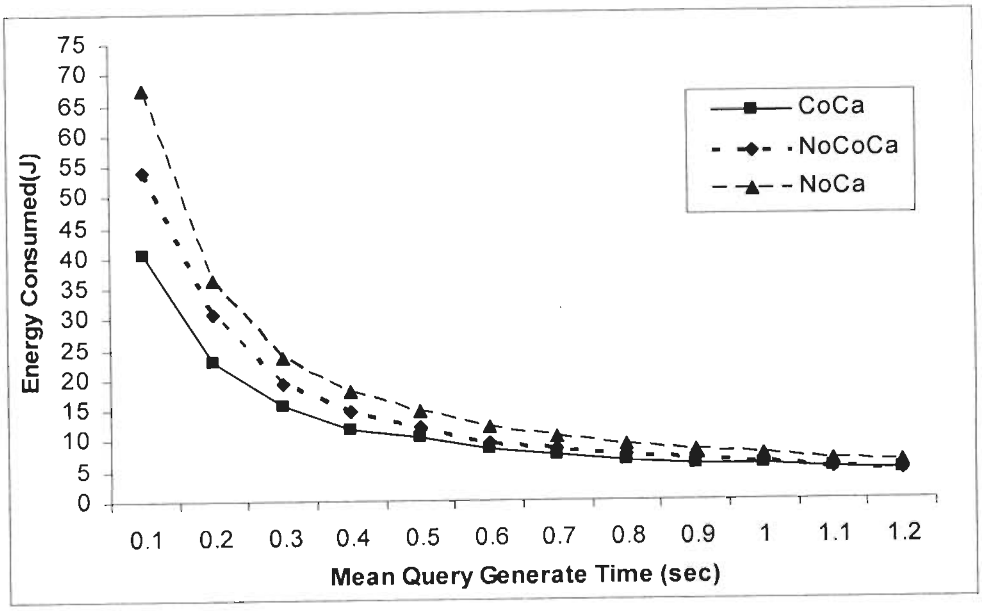


Figure 6.7(d). Effect of Mean Query Generate Time on Overall Energy Consumed

Figure 6.7(d) shows over all energy consumed by the network by CoCa, NoCoCa and NoCa schemes. In each scheme, it includes energy consumed by all the DNs on data/query path including sink and event sensing nodes. When T_{mean} is relatively very small compared to *TTL* values of data items, energy consumed by CoCa is 40% less than energy consumed by NoCa for a cache size of 5 data items. Also, for same cache size NoCoCa consumes 22% less energy than energy consumed by NoCa.

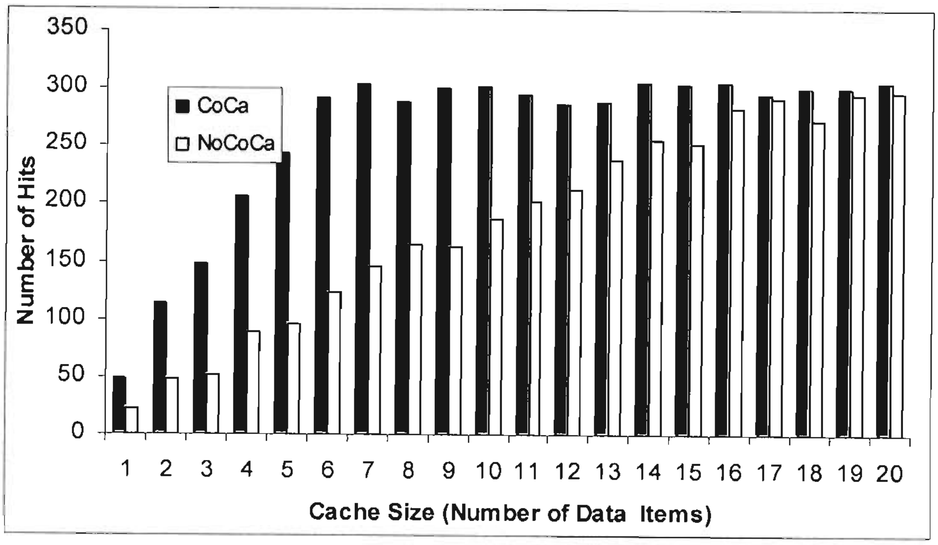


Figure 6.8(a). Effect of Cache Size on Cache Hits

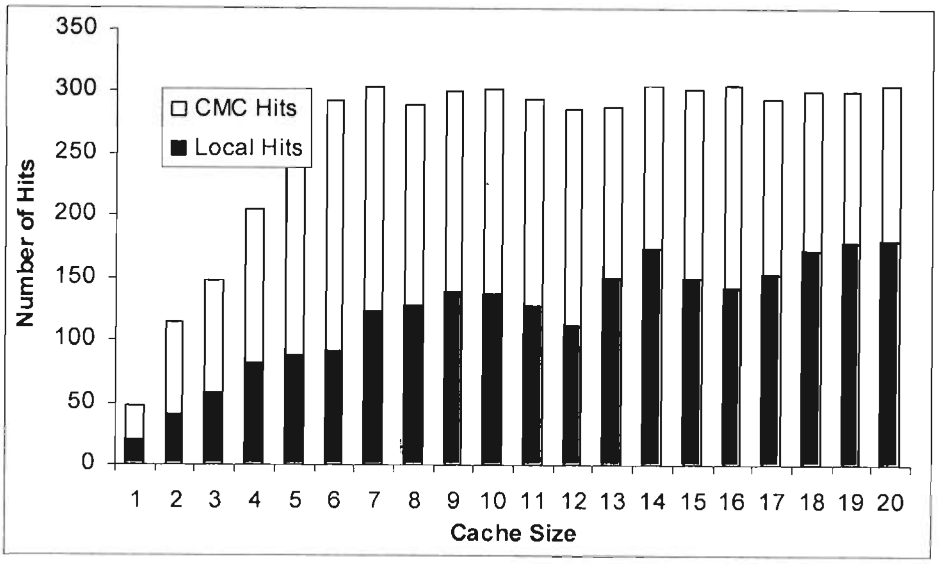


Figure 6.8(b). Effect of Cache Size on Local and CMC Hits in CoCa

6.4.3.2 Effects of Cache Size

Figures 6.8(a), 6.8(b), 6.8(c) and 6.8(d) show the effects of cache size on performance metrics. T_{mean} is kept fixed at 0.2 seconds and cache size is varied to observe its effects on cache hits, data item replacements and overall energy consumed by the network. Initially, CoCa shows sharp rise in number of cache hits with increasing cache size but saturates after certain limit and increasing cache size beyond that does not increase cache hits any more. In Figure 6.8(a), increasing cache size above 7 does not improve upon number of cache hits in CoCa. However, in NoCoCa rise in number of cache hits is not as sharp as in CoCa and also saturates later at cache size 11 or 12. This is

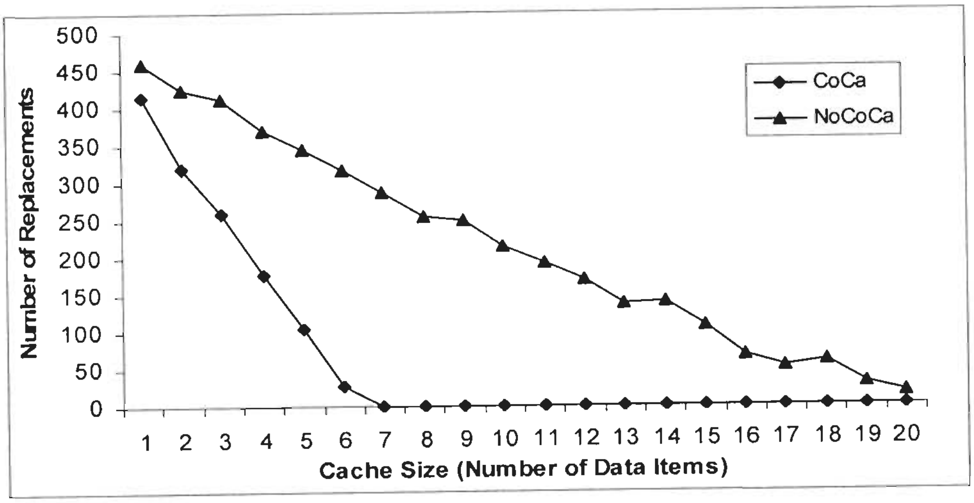


Figure 6.8(c). Effect of Cache Size on Number of Data Item Replacements

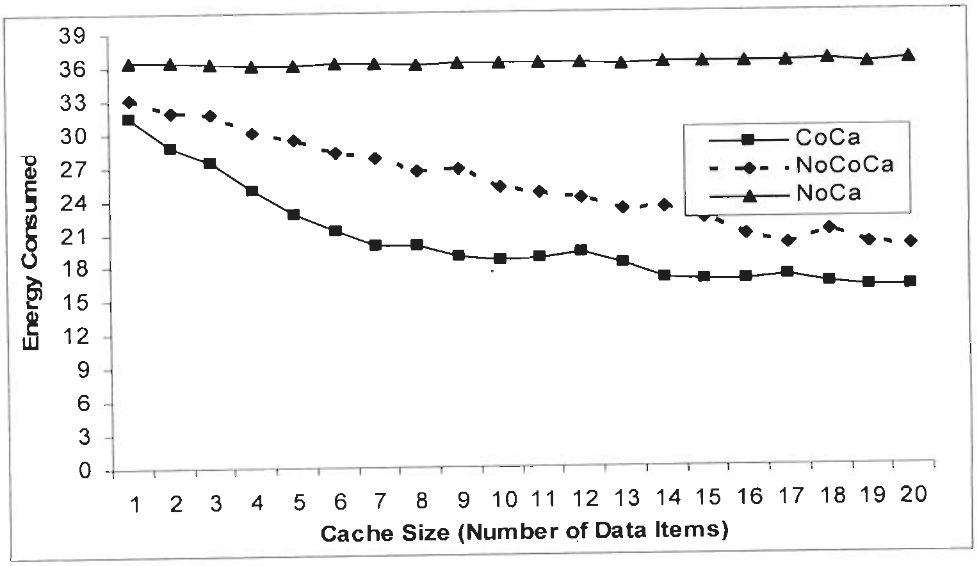


Figure 6.8(d). Effect of Cache Size on Overall Energy Consumed

due to the fact that for same cache size, DN in CoCa has more cache storage locations (i.e. its own local cache plus local caches of all cache nodes in its CAZ) as compared to DN in NoCoCa where it uses only its own local cache. Beyond a particular cache size, cache hits do not increase because after certain time *TTL* values of many data items in cache start expiring making them stale. Figure 6.8(b) shows contribution of local hits and CMC hits to total cache hits achieved in CoCa.

Figure 6.8(c) shows the impact of cache size on number of data item replacements both in CoCa and NoCoCa. As number of cache locations per DN in CoCa is much higher than locations in NoCoCa, hence number of data item replacements for CoCa is much lower as compared to NoCoCa and falls much sharply with increasing cache size. Figure 6.8(d) gives impact of cache size on overall energy consumption by network in CoCa and

NoCoCa. This is also compared with scenario where no caching is used (NoCa). With very low cache size the difference in overall energy consumed by the network in CoCa and NoCoCa is less, but with increase in cache size the difference is quite significant. In comparison to NoCa, CoCa consumes 60% less energy and NoCoCa consumes up to 47% less energy.

But for very large cache size performance of CoCa and NoCoCa again starts approaching close to each other. This is due to the fact that due to large number of storage locations in local cache, many data items start becoming stale before local cache is full and thus freeing the location for new data item storage. Hence in CoCa, DN acting as ACM caches data repeatedly in local cache and very infrequently uses CMC. Therefore, CoCa starts behaving like NoCoCa at very high cache size. But we know SNs have limited memory to be spared as cache, therefore CoCa scheme is suited most for WSN deployments.

6.5 Conclusions

Motivated by several WSN applications where huge amount of data is generated at active nodes and sink issues queries randomly to fetch these data items, present work targets at caching maximum data items in network. Proposed scheme exploits cooperation among SNs to realize efficient caching scheme. Entire sensor field is converted into a grid of square sized cells and a query/data dissemination path is set between a pair of sink and event. Some nodes on this path form cooperative zones in which SNs cooperate among themselves to realize a much larger CMC than individual SN's storage. Low power radio is used for communication among nodes in a zone for any cache management activity. Proposed caching scheme ensures maximum data availability to sink and that too in its close proximity. Also, proposed caching scheme significantly enhances performance of the network in terms of lower overall energy consumption and fewer data item replacements needed. As part of complete cooperating caching strategy, a cache discovery scheme; token based cache admission control; single copy caching rule; and least utility based item replacement policy are developed. Simulation results show that data dissemination scheme that uses proposed *CC* provides significant gains in performance compared to data dissemination schemes where no *CC* is used or no caching is used at all.

Chapter 7

Conclusions and Scope for Future Work

This chapter concludes the thesis and discusses future scope of the work.

7.1 Conclusions

Main focus of our research has been to prolong operational lifetime of a WSN by minimising inter-node transmissions while disseminating voluminous data from source to sink. Various schemes to reduce inter-node transmissions have been proposed and evaluated for various WSN scenarios.

The major contributions of the reported work are outlined as follows:

1. A in-network data processing approach called as Two-Way Sliding Window (TWSW) data filtering is developed to block most of undesired data readings nearer to the source. The filtering scheme exploits the fact that large numbers of SNs sensing a common event over a very long period generate sensed data which includes various redundant and spurious readings. Also, there exists some spatial and/or temporal correlation among various generated data items which forms the base for design of filtering function used by the scheme.

TWSW data filtering uses a filtering window which implements a filtering function with two types of filtering, namely redundant data filtering (RDF) and spurious data filtering (SDF). TWSW data filtering window is used both at CNs and at CHNs. It accomplishes following tasks:

i. RDF blocks redundant data at CNs and CHNs. Redundant data counts for most of unnecessary/undesired inter-node transmissions in a WSN. Based on required granularity of data at sink, RDF sets a filtering window both at CNs and CHNs with appropriate upper and lower limits. This window allows only those data readings to cross for further dissemination which vary from previously disseminated data readings by an amount which is differentiable at sink. Rest of data readings are treated as redundant and hence are blocked.

ii. SDF isolates and blocks spurious data. Spurious data is the data whose value falls very much outside the limits of data expected from even most erratic behaviour of an event. Such data may be generated due to many reasons like software/hardware malfunctioning, caused by adversary or temporary obstruction caused in a channel etc. SDF filters such data by blocking its dissemination towards sink at active CNs and also at CHNs.

iii. TWSW data filtering function when used at CHNs with slight modification, also detects '*zone effect*'. Zone effect is a situation where few active CNs are under the influence of temporary abnormal condition not caused by the event, but is the result of some external condition. However, the rate of change in sensed parameter observed at active CNs in this region is such that readings generated pass both RDF and SDF checks i.e. data values fall within valid bounds of both RDF and SDF. But, the rate of change in successive data readings generated is much different from rest of the nodes under normal conditions. This results in mean of filtering windows at these CNs to drift significantly away from rest of CNs without being noticed locally. Hence, TWSW data filtering when used at CHNs takes some broader view of sensor field as it receives readings from multiple CNs and hence is able to detect zone effect.

TWSW data filtering in a WSN results in significant percentage of average energy saving compared to energy consumed by a dissemination scheme where no data filtering is used. Percentage energy saving is 19.8% when number of CNs are 10 and 25.8% when number of CNs are 100.

2. A novel Adaptive Sensing and Dynamic Data Reporting (ASDDR) scheme is developed that solves the problem of adapting to event behaviour by dynamically adjusting the sensing frequency at active CNs. ASDDR solves above problem by making CHNs to monitor the flow of data reports received from CNs in their respective clusters. With in a stipulated time CHN counts the numbers of data reports received after TWSW filtering check at CNs and infers the current behaviour of the event. Inference is drawn by comparing the number of reports received with suitable upper and lower limits decided by the proposed algorithm for that node. Decision is local to the cluster and new value of

CNSI (CNDI and CHDI thereof) for current cluster is computed. CNSI is broadcasted with in the respective cluster to convey new sensing frequency to active CNs in it.

Main features of ASDDR are:

i. If event becomes more dynamic, sensing frequency is increased and if event becomes relatively static, sensing frequency is decreased. Increment/decrement value in sensing interval is in small steps and is dependent on application as well as event behaviour.

ii. For relatively less random behaviour of an event, ASDDR gives best performance in terms of overall network energy savings achieved and at the same time transfers same information to sink with less data as compared to no data filtering or TWSW data filtering. At very high rate of change in event behaviour (randomness) it may consume more network energy than either of no data filtering case or TWSW data filtering. However, it captures every possible interesting observation about event. Saving network energy at the cost of lost accuracy or wrong inference is seriously damaging and unwanted situation for most applications.

iii. Another important feature of ASDDR scheme is that average delay reduces with increased randomness which is quite interesting. With more randomness more observations about event are generated at CNs which must be disseminated towards sink. It seems that longer data queues will be formed at CNs and CHNs leading to more delays. ASDDR however uses a dissemination model which responds to changed sensing frequency by adjusting dissemination intervals at CNs and CHNs in such a manner that average delay gets reduced. Application may impose certain upper and lower limit on the reporting time for CHNs and between these limits ASDDR can easily adjust sensing and dissemination frequencies.

ASDDR approach is very simple and easy to implement although very effective as revealed by simulation study. Percentage energy savings achieved using ASDDR with respect to no data filtering (LEACH) case ranges from 46.4% with 10 CNs to 58.1% with 100 CNs. This is much higher than savings achieved when TWSW is used with fixed sensing frequency, which ranges from 37.7% with 10 CNs to 43.8% with 100 CNs.

3. A virtual grid construction and data dissemination scheme named as Grid Based Data Dissemination (GBDD) is devised. GBDD constructs virtual grid over entire sensor field which results in achieving more energy efficiency than existing approaches. GBDD

exploits location awareness and dual radio mode of SNs to build virtual grid structure over entire sensor field. GBDD has following main features:

i. Unlike TTDD, where each event appearing in sensor field starts a separate grid construction, in GBDD only first sink appearing in the sensor field triggers grid construction process. Virtual grid with sufficiently large lifetime is created and is shared by all other sinks appearing during that period without the need for constructing new grid as created in TTDD for every appearance of event. New grid construction is triggered only if no valid grid is present when a sink appears. Therefore, a lot of energy spent in grid construction is saved.

ii. GBDD establishes path between source-sink pair which follows smallest possible path and hence consumes very less network energy as compared to other schemes. Due to smaller cell size and knowledge of upstream DN at each step, path is almost straight line from source to sink.

iii. GBDD is also very effective in handling sink and event movements. Movements are efficiently managed through local interactions (message passing) among neighbours and path sharing. However, if sink and/or event moves with very high speed, due to smaller cell size GBDD may consume more energy.

iv. Apart from other aims and advantages, GBDD is devised with the objective to make it tailor made for Cooperative Caching (CC) scheme also developed as part of current study.

On an average, GBDD shows up to 43% overall network energy saving as compared to TTDD. Also, GBDD shows 30% improvement in average delay computed across all source-sink pairs for a data packet to reach from source to sink. However, for very high sink/event speeds, the average packet delay increases more in GBDD due to smaller cell size which needs frequent reestablishment of partial paths..

4. A cooperative caching (CC) scheme is developed which addresses two basic issues, namely excess data flow handling and cutting down query traversal path to reduce inter-node transmissions. Some designated nodes on data/query path act as coordinators for forming cooperative zones around them. All nodes in such zones cooperate among themselves to realize a much larger cumulative cache (CMC) than individual node's storage. Low power radio is used for communication among nodes in a cooperative zone for all coordination and cache management activities. CC ensures maximum data availability to sink and that too in its close proximity. Also, CC significantly enhances

performance of the network in terms of lower overall energy consumption by avoiding repetitive event sensing and by reducing path traveled by query issued from sink. As part of the complete CC scheme, following modules are developed:

i. *Cache Discovery Scheme*: It effectively locates and fetches required data item from with in the network (if cached) with minimum time and if no cached copy is found, it initiates a *FDSA* procedure that aggregates data item at source node after freshly sensing the event.

ii. *Token based Cache Admission Control:* A cache admission control is developed that uses a special token based approach to decide the location for caching a data item. It increases data availability nearer to sink and avoids unnecessary replication of a data item by allowing only those nodes that hold special token (*CT*) to cache or replace data item. These nodes are known as cache managers. Cache managers are selected by passing CT among candidate nodes (i.e. nodes that have more than some minimum numbers of locations free) on data/query path(s) in such a manner that a candidate node nearest to sink has the highest possibility to receive CT among all candidate nodes.

iii. *Single Copy Cache Rule: TTL* based weak cache consistency model is used with a caching rule which ensures that for a source-sink pair only a single copy of the data item is maintained in the network.

iv. *Least Utility based Item Replacement Policy*: In case of required item replacement, an item with least utility is evicted out. Utility of an item is determined based upon its access rate and its *TTL* value. An item gets replaced only from the cache of the node which is acting as cache manager i.e. node that holds *CT*.

For very small mean query generate time and normal cache size (accommodating 5 data items), data dissemination scheme employing cooperative caching consumes up to 60% less than energy than consumed when no caching scheme is used.

7.2 Scope for the Future Work

There is lot of scope for extending the current study to achieve energy efficiency in WSNs. Some of the future directions are:

1. Due to various reasons like dense deployment of SNs in sensor field, multiple SNs located at different locations sensing common event during a particular period and multiple parameters about event(s) being sensed etc, there exist complex spatio-

temporal correlation among data items accumulated at a node. Due to lack of global clock synchronization among SNs, entire sensed data accumulated may not pertain to exactly same time. Present study has focused on devising data filtering strategy rather than solving intricacies of spatio-temporal correlation among data items. Whereas, in future this work can be extended by setting filtering window mean and bounds based on values found by exploiting such relations among data items and thus TWSW data filtering and ASDDR can be adapted to any application and sensing scenario more effectively.

2. Self adaptation to changed conditions is seen as one of the most desirable feature for autonomous operation of a WSN. Proposed ASDDR makes SNs to adapt to event behaviour by dynamically altering sensing frequency, however the sizes of filtering windows remain same both for RDF and SDF. Due to multiple sensors onboard, SNs can be made to sense various event parameters during same or different observation periods. Also, if SNs are recoverable, they can be deployed at varying locations where conditions are quite different. In either of scenarios above, filtering window as used in TWSW must be set by SNs themselves by reacting to changed parameter to be sensed and/or changed environmental conditions. Hence, work on ASDDR can be extended in future in a manner so that it also sets bounds of filtering window (for both RDF and SDF) dynamically by adapting to new scenario.
3. Limited mobility of sinks and/or events in WSNs can be handled by many techniques proposed in literature and also our proposed GBDD scheme has proved very effective for this purpose. However, many WSN scenarios are emerging where apart from mobile sink and event, field SNs are mobile. Topology management tasks like forming virtual grid, forming clusters and establishing/maintaining paths between source-sink pairs become extremely challenging in the face of mobile SNs. Therefore, in future work related to GBDD can be extended to handle such SNs' movements.
4. Proposed caching assumes that path between source node and sink remains fixed for entire observation period and nodes acting as DNs do not change. But, in many scenarios either due to mobility of SNs as stated above or due to load balancing, the path between same set of source-sink pair may change over time. Therefore, in future proposed CC may be modified to adapt to such changed paths.

5. In proposed CC, caching zones may be spread across entire WSN due to multiple source-sink pairs. Data retrieval is query based where in response to queries from sink, data items move towards it and are interpreted there. Whereas one can observe that a cache zone holds closely related data items. Hence, instead of moving bulky data towards sink, computations on data can be performed at few selected nodes with in caching zone(s) itself and only smaller resultant information may be transferred to sink. In this way, a significant amount of reduction in network energy can be achieved due to reduced inter-node transmissions. Therefore, in future proposed GBDD and CC can be used to device software agent type of data dissemination approach by eliminating the need to move entire cached data to sink.

References

[1] A. Anand, S. Sachan, K. Kapoor and S. Nandi, "QDMAC: An Energy Efficient Low Latency MAC Protocol for Query Based Wireless Sensor Networks," *Proceedings of the 10th International Conference on Distributed Computing and Networking (ICDCN)*, pp. 306-317, January 2009.

[2] A. Ephremides, J.E. Wieselthier and D.J. Baker, "A Design Concept for Reliable Mobile Radio Networks with Frequency Hopping Signaling," *Proceedings of the IEEE*, Vol. 75, No. 1, pp. 56–73, 1987.

[3] A. Mainwaring, D. Culler, J. Polastre, R. Szewczyk and J. Anderson, "Wireless Sensor Networks for Habitat Monitoring," *Proceedings of the 1st ACM international workshop on Wireless sensor networks and applications*, pp. 88–97, 2002.

[4] A. Manjeshwar and D.P. Agarwal, "TEEN: A Routing Protocol for Enhanced Efficiency in Wireless Sensor Networks," *Proceedings of the 15th International Parallel and Distributed Processing Symposium*, pp. 2009-2015, April 2001.

[5] A. Manjeshwar and D.P. Agrawal, "APTEEN: A Hybrid Protocol for Efficient Routing and Comprehensive Information Retrieval in Wireless Sensor Networks," *Proceedings of the Parallel and Distributed Processing Symposium (IPDPS)*, pp. 195-202, April 2002.

[6] A. McDonald and T. Znati, "A Mobility-Based Framework for Adaptive Clustering in Wireless Ad Hoc Networks," *IEEE Journal on Selected Areas in Communications (JSAC)*, Vol. 17, No. 8, pp. 1466–1486, August 1999.

[7] A. Perrig, R. Canetti, J.D. Tygar and D. Song, "The TESLA Broadcast Authentication Protocol," *RSA CryptoBytes*, Vol. 5, 2002.

[8] A. Perrig, R. Szewczyk, V. Wen, D. Culler and J.D. Tygar. "SPINS: Security Protocols for Sensor Networks," *Wireless Networks*, Vol. 8, Issue 5, pp. 521 – 534, 2002.

[9] A. Roumy and D. Gesbert. "Optimal Matching in Wireless Sensor Networks," *IEEE Journal on Selected Topics in Signal Processing*, Vol.1, No. 4, December 2007.

[10] A. Rousskov and D. Wessels, "Cache Digests," *Computer Networks and ISDN Systems*, Vol. 30, No. 22–23, pp.2155–2168, 1998.

[11] A. Youssef, M. Younis, M. Youssef and A. Agrawala, "Distributed Formation of Overlapping Multi-Hop Clusters in Wireless Sensor Networks," *Proceedings of the 49th Annual IEEE Global Communication Conference (Globecom'06)*, pp. 1-6, November- December 2006.

[12] A.G. Ruzzelli, G. O'Hare, R. Jurdak and R. Tynan, "Advantages of Dual Channel MAC for Wireless Sensor Networks," *Proceedings of the 1st International Conference on Communication System Software and Middleware (Comsware 2006)*, pp. 1-3, 2006.

[13] A.S. Sairam and G. Barua, "Bandwidth Management Using Load Balancing," *Proceedings of the First International Conference on Communication System Software and Middleware (COMSWARE 2006)*, January 8-12, 2006.

[14] B. Chen, K. Jamieson, H. Balakrishnan and R. Morris, "SPAN: An Energy-Efficient Coordination Algorithm for Topology Maintenance in Ad Hoc Wireless Networks," *Wireless Networks*, Vol. 8, No. 5, pp. 481–94, September 2002.

[15] B. Dey, S. Akhil, and S. Nandi, "Clustering Problem in Large-Scale Sensor Network: Possibilities and Requirements," *Proceedings of the International Conference on Advanced Communication Systems (ICACS-2007)*, January 2007.

[16] B. Krishnamachari, D. Estrin and S. Wicker, "Modeling Data Centric Routing in Wireless Sensor Networks," *Proceedings of the IEEE INFOCOM*, June 2002.

[17] C. Intanagonwiwat, R. Govindan, D. Estrin, J. Heidemann and F. Silva, "Directed Diffusion for Wireless Sensor Networking," *IEEE/ACM Transactions on Networking*, Vol. 11, No. 1, pp. 2 – 16, February 2003.

[18] C. Schurgers and M.B. Srivastava, "Energy Efficient Routing in Wireless Sensor Networks," *IEEE Communications for Network-Centric Operations: Creating the Information Force (MILCOM)*, Vol. 1, pp. 357-361, 2001.

[19] C. Schurgers, V. Tsiatsis, S. Ganeriwal and M. B. Srivastava, "Topology Management for Sensor Networks: Exploiting Latency and Density," *Proceedings of the 3rd ACM international symposium on Mobile ad hoc networking & computing (MobiHoc)*, pp. 135-145, 2002.

[20] C. Wang, K. Sohraby, B. Li, M. Daneshmand and Y. Hu, "A Survey of Transport Protocols for Wireless Sensor Networks," *IEEE Network*, Vol. 20, No. 3, pp. 34–40, 2006.

[21] C.R. Lin and M. Gerla, "Adaptive Clustering for Mobile Wireless Networks," *IEEE Journal on Selected Areas in Communications*, Vol.15, No. 7, pp. 1265–1275, 1997.

[22] C.R. Lin and M. Gerla, "Adaptive Clustering for Mobile Wireless Networks," *IEEE Journal on Selected Areas in Communications*, Vol.15, No. 7, pp.1265–1275, 1997.

[23] C.Y. Chow, H.V. Leong and A.T.S. Chan, "GroCoca: Group-Based Peer-To-Peer Cooperative Caching in Mobile Environment," *IEEE Journal on Selected Areas in Communications*, Vol. 25, No. 1, pp. 179-191, 2007.

[24] D. Adam, V. Thiemo, A. Juan and R. Hartmut, "Distributed TCP Caching for Wireless Sensor Networks," *Proceedings of the 3rd Annual Mediterranean Ad Hoc Networking Workshop (MedHocNet 2004)*, June 2004.

[25] D. Braginsky and D. Estrin, "Rumor Routing Algorithm for Sensor Networks," *Proceedings of the First ACM Workshop on Sensor Networks and Applications (WSNA)*, pp. 22-31, October 2002.

[26] D. Estrin, L. Girod, G. Pottie and M. Srivastava, "Instrumenting the World with Wireless Sensor Networks," *Proceedings of the International Conference on Acoustics, Speech and Signal Processing (ICASSP 2001)*, Vol.4, pp. 2033-2036, 2001.

[27] D. Katsaros and Y. Manolopoulos, "Caching in Web Memory Hierarchies," *Proceedings of the ACM symposium on applied computing*, pp. 1109–1113, 14–17 March 2004.

[28] D. Katsaros and Y. Manolopoulos, "Web Caching in Broadcast Mobile Wireless Environments," *IEEE Internet Computing*, Vol. 8, No. 3, pp.37–45, 2004.

[29] D. McIntire, K.H. Hing, B. Yip, A. Singh, W. Wu and W. Kaiser, "The Low Power Energy Aware Processing (LEAP) System," *Proceedings of the 5th International Conference on Information processing in sensor networks*, pp. 449-457, April 2006.

[30] D. Nikos, K. Dimitrios and M. Yannis, "Cooperative Caching in Wireless Multimedia Sensor Networks," *Mobile Networks and Applications*, Vol. 13, No. 3-4, pp. 337-356, 2008.

[31] D. Wessels and K. Claffy, "ICP and the Squid Web cache," *IEEE Journal on Selected Areas in Communications,* Vol. 16, No. 3, pp.345–357, 1998.

[32] D.E. Cooper, P. Ezhilchelvan and I. Mitrani, "A Family of Encounter-Based Broadcast Protocols for Mobile Ad-Hoc Networks," *Proceedings of the1st International Workshop of the EURO-NGI Network of Excellence*, pp. 235-248, June 2004.

[33] D.E. Cooper, P. Ezhilchelvan and I. Mitrani, "High Coverage Broadcasting for Mobile Ad Hoc Networks," *Proceedings of the 3rd International IFIP-TC6 Networking Conference*, pp. 100-111, May 2004.

[34] D.J. Baker and A. Ephremides, "The Architectural Organization of A Mobile Radio Network via A Distributed Algorithm," *IEEE Transactions on Communications*, Vol. 29, No. 11, pp. 1694–1701, 1981.

[35] D.J. Baker, A. Ephremides and J.A. Flynn, "The Design and Simulation of A Mobile Radio Network with Distributed Control," *IEEE Journal on Selected Areas in Communications*, Vol. 2, No.1, pp. 226–237, 1984.

[36] E. Fasolo, M. Rossi, J. Widmer and M. Zorzi, "In-Network Aggregation Techniques for Wireless Sensor Networks: A Survey," *IEEE Wireless Communications*, Vol. 14, No. 2, pp. 70-87, April 2007.

[37] E. Shih, P. Bahl and M.J. Sinclair, "Wake on Wireless: An Event Driven Energy Saving Strategy for Battery Operated Devices," *Proceedings of the 8th annual International Conference on Mobile computing and networking (Mobicom, 2002)*, pp. 160-171, 2002.

[38] E.A. Brewer, R.H. Katz, Y. Chawathe, S.D. Gribble, T. Hodes, M. Giao Nguyen Stemm, T. Henderson, E. Amir, H. Balakrishnan, A. Fox, V.N. Padmanabhan and S. Seshan, "A Network Architecture for Heterogeneous Mobile Computing," *IEEE Personal Communications*, Vol. 5, No. 5, pp. 8-24, 1998.

[39] F. Bonnet, P. Ezhilchelvan and E. Vollset, "Quiescent Consensus in Mobile Ad-Hoc Networks Using Eventually Storage-Free Broadcasts," *Proceedings of the*

21st annual ACM Symposium on Applied Computing, pp 670-674, April 2006.

[40] F. Sailhan and V. Issarny, "Cooperative Caching in Ad Hoc Networks," *Proceedings of the International Conference on Mobile Data Management (MDM),* pp. 13–28, 2003.

[41] F. Sailhan and V. Issarny, "Energy-Aware Web Caching for Mobile Terminals," *Proceedings of the IEEE international conference on distributed computing systems workshops (ICDCSW),* pp. 820–825, 2002.

[42] F. Ye, A. Chen, S. Lu, L. Zhang, "A Scalable Solution to Minimum Cost Forwarding in Large Sensor Networks," *Proceedings of the 10th International. Conference on Computer Communication and Networks*, pp. 304–09, 2001.

[43] G. Hariharan, S. Mehta and S.K. Gupta, "Location Based Information Retrieval Framework," *Proceedings of the SPIE Data Mining Conference*, pp. 235-143, April 2003.

[44] G. Hartl and B. Li, "infer: A Bayesian Approach towards Energy Efficient Data Collection in Dense Sensor Networks," *Proceedings of the 25th IEEE International Conference on Distributed Computing Systems (ICDCS 2005),* pp. 371-380, June 2005.

[45] G.J. Pottie and W.J. Kaiser, "Wireless Integrated Network Sensors," Communications of the ACM, Vol. 43, No. 5, pp. 51–58, May 2000.

[46] H. Albert, R. Kravets and I. Gupta, "Building Trees Based on Aggregation Efficiency in Sensor Networks," *Elsevier Ad Hoc Networks,* Vol. 5, No. 8, pp. 1317-1328, 2007.

[47] H. Chan and A. Perrig, "ACE: An Emergent Algorithm for Highly Uniform Cluster Formation," *Proceedings of the 1st European Workshop on Sensor Networks (EWSN)*, 154-171, January 2004.

[48] H. Chan, M. Luk and A. Perrig, "Using Clustering Information for Sensor Network Localization," *Proceedings of the IEEE International Conference on Distributed Computing in Sensor Systems, (DCOSS'05)*, June 2005.

[49] H. Che, Y. Tung and Z. Wang, "Hierarchical Web Caching Systems: Modeling, Design and Experimental Results," *IEEE International Journal on Selected Areas in Communications,* Vol. 20, No. 7, pp. 1305–1314, September 2002.

[50] H. Cheng, Q. Liu and X. Jia, "Heuristic Algorithms for Real-Time Data Aggregation in Wireless Sensor Networks," *Proceedings of the ACM International Conference on Communications and Mobile Computing*, pp. 1123-1128, July 2006.

[51] H. Gupta, V. Navda, S. R. Das and V. Chowdhary, "Efficient Gathering of Correlated Data in Sensor Networks," *ACM Transactions on Sensor Networks (TOSN)*, Vol. 4, No. 1, January 2008.

[52] H. Jun, M. Ammar, M. Corner and E. Zegura, "Hierarchical Power Management in Disruption Tolerant Networks with Traffic-Aware Optimization," *Proceedings of the ACM SIGCOMM workshop on Challenged networks*, pp. 245-252, 2006.

[53] H. Luo, F. Ye, J. Cheng, S. Lu and L. Zhang, "TTDD: Two-Tier Data Dissemination in Large-Scale Wireless Sensor Networks", *Kluwer Academic Publishers' Wireless Networks,* Vol. 11, No. ½, pp. 161-175, January 2005.

[54] H. Shen, S. Das, M. Kumar and Z. Wang, "Cooperative Caching with Optimal Radius in Hybrid Wireless Networks," *Proceedings of the IFIP International Federation for Information Processing 2004(NETWORKING 2004, LNCS 3042)*, pp. 841–853, May 2004.

[55] H.B. Lim, Y.M. Teo, P. Mukherjeel, V.T Lam, W.F. Wong and S. See, "Sensor Grid: Integration of Wireless Sensor Networks and The Grid," *Proceedings of the 30th IEEE Conference on Local Computer Networks,* pp. 91-98, November 2005.

[56] H.O. Tan and I. Korpeoglu, "Power Efficient Data Gathering and Aggregation in Wireless Sensor Networks," *ACM SIGMOD Record,* Vol. 32, No. 4, pp. 66–71, December 2003.

[57] H.R. Naqvi and L.M. Patnaik, "A Distributed Channel Access Protocol for Ad-hoc Networks with Feedback Power Control," *IEEE Transactions on Mobile Computing*, Vol. 5, No.10, pp.1448-1459, October 2006.

[58] I. Kadayif and M. Kandemir, "Tuning In-Sensor Data Filtering to Reduce Energy Consumption in Wireless Sensor Networks," *Proceedings of the Design, Automation and Test in Europe Conference and Exhibition (DATE'04)*, Vol. 2, pp. 852-857, 16-20 February 2004.

[59] I. Solis and K. Obraczka, "Isolines: Energy-efficient Mapping in Sensor Networks," *Proceedings of the 10th IEEE Symposium on Computers and*

Communications (ISCC 2005), pp. 379-385, June 2005.

[60] I. Stojmenovic and X. Lin, "GEDIR: Loop-Free Location Based Routing in Wireless Networks," *Proceedings of the International Conference on Parallel and Distributed Computing and Systems,* November 3–6 1999; *IEEE Transactions on Parallel Distributed Systems*, Vol. 12, No. 10, pp. 1023-1032, 2001.

[61] I. Stojmenovic, M. Seddigh and J. Zunic, "Dominating Sets and Neighbor Elimination-Based Broadcasting Algorithms in Wireless Networks," *IEEE Transactions on Parallel and Distributed Systems*, Vol. 13, No. 1, pp. 14–25, 2002.

[62] I.F. Akyildiz, W. Su, Y. Sankarasubramaniam and E. Cayirci, "A Survey on Sensor Networks," *IEEE Communications Magazine*, Vol. 40, No. 8, pp.102-114, 2002.

[63] I.F. Akyildiz, W. Su, Y. Sankarasubramaniam and E. Cayirci, "Wireless Sensor Networks: A Survey," *Computer Networks*, Vol. 38, No. 4, pp. 393–422, March 2002.

[64] J. Choi, J.W. Lee, K. Lee, S. Choi, W.H. Kwon and H.S. Park, "Aggregation Time Control Algorithm for Time Constrained Data Delivery in Wireless Sensor Networks," *Proceedings of the 63rd IEEE Vehicular Technology Conference*, Vol. 2, pp. 563-567, May 2006.

[65] J. Kulik, W.R. Heinzelman and H. Balakrishnan, "Negotiation-Based Protocols for Disseminating Information in Wireless Sensor Networks," *Wireless Networks,* Vol. 8, No. 2/3, pp. 169–85, 2002.

[66] J. Zhang and V. Varadharajan, "A New Security Scheme for Wireless Sensor Networks," *Proceedings of the Global Communications Conference,* pp.128-132, December 2008.

[67] J.N. Al-Karaki and A.E. Kamal, "End-to-End Support for Statistical Quality of Service in Heterogeneous Mobile Ad Hoc Networks," *Computer Communications*, Vol. 28, No. 18, pp. 2119-2132, Nov. 2005.

[68] J.N. Al-Karaki and A.E. Kamal, "Routing Techniques in Wireless Sensor Networks: A Survey," *IEEE Wireless Communications*, Vol. 11, issue 6, pp. 6-

28, December 2004.

[69] J.N. Al-Karaki, R. Ul-Mustafa and A.E. Kamal, "Data Aggregation in Wireless Sensor Networks: Exact and Approximate Algorithms," *Proceedings of the IEEE Workshop on High Performance Switching and Routing (HPSR 2004)*, pp. 241-245, April 2004.

[70] K. Chowdhury, N. Nandiraju, D. Cavalcanti and D.P. Agrawal, "C- MAC – A Multi-Channel Energy Efficient MAC for Wireless Sensor Networks," *Proceedings of the IEEE Wireless Communications and Networking Conference (WCNC)*, pp. 1172-177, April 2006.

[71] K. Dasgupta, K. Kalpakis and P. Namjoshi, "An Efficient Clustering Based Heuristic for Data Gathering and Aggregation in Sensor Networks," *IEEE Wireless Communications and Networking (WCNC 2003)*, Vol. 3, pp. 1948-1953, March 2003.

[72] K. Naoto and L. Shahram, "A Survey on Data Compression in Wireless Sensor Networks," *Proceedings of the International Conference on Information Technology: Coding and Computing (ITCC'05)*, Vol. 2, pp. 8-13, April 2005.

[73] K. Prabh and T. Abdelzaher, "Energy-Conserving Data Cache Placement in Sensor Networks," *ACM Transactions on Sensor Networks,* Vol. 1, No. 2, pp. 178–203, 2005.

[74] K. R¨omer, "Time Synchronization in Ad Hoc Networks," *Proceedings of the ACM Symposium on Mobile Ad-Hoc Networking and Computing*, pp. 173-182, October 2001.

[75] K. R¨omer. "Temporal Message Ordering in Wireless Sensor Networks", *Proceedings of the IFIP Mediterranean Workshop on Ad-Hoc Networks*, pp. 131–142, June 2003.

[76] K. Sohrabi and J. Pottie, "Protocols for Self-Organization of a Wireless Sensor Network," *IEEE Personal Communications*, Vol. 7, No. 5, pp. 16–27, 2000.

[77] K. Xu and M. Gerla, "A Heterogeneous Routing Protocol Based on A New Stable Clustering Scheme," *Proceedings of the IEEE Military Communications Conference (MILCOM 2002),* Vol. 2, pp. 838-843, October 2002.

[78] L. Fan, P. Cao, A.Z. Almeida and J.M. Broder, "Summary Cache: A Scalable Wide-Area Web Cache Sharing Protocol," *IEEE/ACM Transactions on Networking*, Vol. 8, No. 3, pp.281–293, 2000.

[79] L. Yin and G. Cao, "Supporting Cooperative Caching in Ad Hoc Networks," *IEEE Transactions on Mobile Computing*, Vol. 5, No. 1, pp.77–89, 2006.

[80] L. Yin and G. Cao, "Supporting Cooperative Caching in Ad Hoc Networks," *IEEE Transactions on Mobile Computing*, 5(1): pp. 77-89, 2006.

[81] Li (Erran) Li and J.Y. Halpern, "A Minimum-Energy Path-Preserving Topology-Control Algorithm," *IEEE Transactions on Wireless Communications*, Vol. 3, No. 3, May 2004.

[82] M. Agarwal and G. Barua, "Caching of Routes in Ad Hoc On-Demand Distance Vector Routing for Mobile Ad Hoc Networks," *Proceedings of the 15th International Conference on Computer Communication (ICCC2002)*, pp. 768-778, August 2002.

[83] M. Ali, T. Suleman and Z.A. Uzmi, "MMAC: A Mobility-Adaptive, Collision-Free Mac Protocol for Wireless Sensor Networks," *Proceedings of the 24th IEEE International Performance, Computing, and Communications Conference*, pp. 401-407, April 2005.

[84] M. Cardei, S. Yang and J. Wu, "Algorithms for Fault-Tolerant Topology in Heterogeneous Wireless Sensor Networks," *IEEE Transactions on Parallel and Distributed Systems*, Vol. 19, No. 4, 545 - 558, April 2008.

[85] M. Chu, H. Haussecker and F. Zhao, "Scalable Information Driven Sensor Querying and Routing for Ad Hoc Heterogeneous Sensor Networks," *International Journal of High Performance Computing Applications*, Vol. 16, No. 3, August 2002.

[86] M. Demirbas, A. Arora and V. Mittal, "FLOC: A Fast Local Clustering Service for Wireless Sensor Networks," *Proceedings of the Workshop on Dependability Issues in Wireless Ad Hoc Networks and Sensor Networks (DIWANS'04)*, June 2004.

[87] M. Ding, X. Cheng and G. Xue, "Aggregation Tree Construction in Sensor Networks," *Proceedings of the 58th IEEE Vehicular Technology Conference*

(VTC 2003), Vol. 4, pp. 2168-2172, October 2003.

[88] M. Marin-Perianu, N. Meratnia, P.J.M. Havinga, L. Moreira S´a de Souza, J. M¨uller, P. Spiess, S. Haller, T. Riedel, C. Decker and G. Stromberg, ”Decentralized Enterprise Systems: A Multi-Platform Wireless Sensor Networks Approach,” *IEEE Wireless Communications,* Vol. 14, No. 6, pp.57–66, 2007.

[89] M. Rahman, S. Hussain, “Effective Caching in Wireless Sensor Network,” *Proceedings of the 21st IEEE International Conference on Advanced Information Networking and Applications Workshops (AINAW'07)*, Vol. 1, pp. 43-47, May 2007.

[90] M. Takaaki and H. Aida, “Cache Data Access System in Ad Hoc Networks,” *Proceedings of the IEEE spring semiannual vehicular technology conference (VTC)*, Vol. 2, pp. 1228–1232, 2003.

[91] M. Yarvis, N. Kushalnagar, H. Singh, A. Rangarajan, Y. Liu and S. Singh, “Exploiting Heterogeneity in Sensor Networks,” *Proceedings of the 24th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM 2005),* Vol. 2, pp. 878-890, March 2005.

[92] M. Ye, C. Li, G. Chen and J. Wu, “EECS: An Energy Efficient Clustering Scheme in Wireless Sensor Networks,” *International Performance, Computing, and Communications Conference, 2005 (IPCCC)*, pp. 535-540, April 2005.

[93] M. Younis, K. Akkaya and A. Kunjithapatham, “Optimization of Task Allocation in A Cluster–Based Sensor Network,” *8th IEEE Symposium on Computers and Communications (ISCC'2003)*, pp. 329-334, June-July 2003.

[94] M.D. Hamilton, P. McKee and I. Mitrani, “Optimal Caching Policies for Web Objects,” *Proceedings of the 9th International Conference High-Performance Computing and Networking (HPCN Europe 2001)*, pp. 94-103, 25-27 June 2001.

[95] N. Bulusu, J. Heidemann and D. Estrin, “GPS-less Low Cost Outdoor Localization for Very Small Devices,” *IEEE Personal Communications Magzine*, Vol. 7, No. 5, pp. 28-34, October 2000.

[96] N. Chand, R.C. Joshi and M. Misra, “Broadcast Based Cache Invalidation and Prefetching in Mobile Environment,” *Proceedings of the International Conference on High Performance Computing (HiPC),* Vol. 3296, pp. 410-419,

2004.

[97] N. Chand, R.C. Joshi and M. Misra, "Cooperative Caching Strategy in Mobile Ad Hoc Networks Based on Clusters," *Springer Wireless Personal Communications*, Vol. 43, No. 1, pp. 41-63, 2006.

[98] N. Chand, R.C. Joshi and M. Misra, "Energy Efficient Cache Invalidation in a Disconnected Wireless Mobile Environment," *International Journal of Ad Hoc and Ubiquitous Computing (IJAHUC)*, Vol. 1, No. 3, pp.184–192, 2005.

[99] N. Jain, R. Biswas, N. Nandiraju and D.P. Agrawal, "Energy Aware Routing for Spatio-Temporal Queries in Wireless Sensor Networks," *IEEE Wireless Communications and Networking Conference (WCNC)*, Vol. 3, pp. 1860-1866, March 2005.

[100] O. Younis and S. Fahmy, "HEED: A Hybrid, Energy-Efficient, Distributed Clustering Approach for Ad Hoc Sensor Networks," *IEEE Transactions on Mobile Computing*, Vol. 3, No. 4, pp. 366–379, 2004.

[101] P. Bahl, A. Adya, J. Padhye and A Wolman. "Reconsidering Wireless Systems with Multiple Radios," *Proceedings of the ACM SIGCOMM Computer Communication Review*, Vol. 34, No. 5, pp. 39-46, 2004.

[102] P. Casari, F. Zorzi, and M. Zorzi. "Efficient Packet Converge–Casting: Relieving the Sink Congestion in Wireless Sensor Networks," *Proceedings of the 18th Annual IEEE International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC'07)*, pp. 1-5, 3-7 September, 2007.

[103] P. Ding, J. Holliday and A. Celik, "Distributed Energy Efficient Hierarchical Clustering for Wireless Sensor Networks," *Proceedings of the IEEE International Conference on Distributed Computing in Sensor Systems (DCOSS'05)*, June 2005.

[104] P. Grover, R Agarwal and A.K. Chaturvedi, "Geolocation Using Transmit and Receive Diversity," *Proceedings of the IEEE GLOBECOM*, pp. 3681-3684, Nov.- Dec. 2004.

[105] P. Kamat, W. Xu, Y. Zhang and W. Trappe, "Temporal Privacy in Wireless Sensor Networks," *Proceedings of the 27th International Conference on Distributed Computing Systems (ICDCS 2007)*, pp. 23, 2007.

[106] P. Nuggehalli, V. Srinivasan, C.F. Chiasserini, "Energy-Efficient Caching Strategies in Ad Hoc Wireless Networks," *Proceedings of the 4th ACM international symposium on Mobile ad hoc networking & computing,* pp. 25–34, 2003.

[107] P. von Rickenbach and R. Wattenhofer, "Gathering Correlated Data in Sensor Networks," *Proceedings of the ACM Joint Workshop on Foundations of Mobile Computing (DIALM-POMC 2004)*, pp. 60-66, October 2004.

[108] P. Wan, K.M. Alzoubi and O. Frieder, "Distributed Construction of Connected Dominating Set in Wireless Ad Hoc Networks," *Mobile Networks and Applications*, Vol. 9, No. 2, pp. 141–149, 2004.

[109] R. Biswas, N. Jain, N. Nandiraju and D.P. Agrawal, "Communication Architecture for Processing Spatio-temporal Continuous Queries in Sensor Networks," *Special issue of the Annals of telecoms devoted to the Sensor Networks*, Vol. 8, No. 7/8, 2005.

[110] R. Cristescu, B. Beferull-Lozano, M. Vetterli and R. Wattenhofer, "Network Correlated Data Gathering with Explicit Communication: NP-Completeness and Algorithms," *IEEE/ACM Transactions on Networking (TON)*, Vol. 14, No. 1, pp. 41-54, February 2006.

[111] R. Ingle and G. Sivakumar, "Tunable Group Key Agreement," *Proceedings of the 32nd IEEE Conference on Local Computer Networks*, pp.1017-1024, October 2007.

[112] R. Kiran Kumar and C. Siva Ram Murthy, "Energy Efficient and Scalable Search in Dense Wireless Sensor Networks," *IEEE Transactions on Computers*, Vol. 58, No. 6, pp. 812-826, June 2009.

[113] R. Kiran Kumar, A. Franklin and C. Siva Ram Murthy, "Coverage Based Expanding Ring Search for Dense Wireless Sensor Networks," *Proceedings of the 15th Annual IEEE International Conference on High Performance Computing (HiPC)*, pp. 245-256, December 17-20, 2008.

[114] R. Krishnan, G. Sivakumar and P. Bhattacharya, "Extracting Decision Trees from Trained Neural Networks," *Journal of Pattern Recognition*, Vol. 32, No. 12, 1999.

[115] R. Nagpal and D. Coore, "An Algorithm for Group Formation in An Amorphous Computer", *Proceedings of the 10th International Conference on Parallel and Distributed Systems (PDCS'98)*, October 1998.

[116] R. Shah and J. Rabaey, "Energy Aware Routing for Low Energy Ad Hoc Sensor Networks," *Proceedings of the IEEE Wireless Communications and Networking Conference (WCNC)*, Vol. 1, pp. 350-355, March 2002.

[117] R.M. Kling, "Intel Motes: Advanced Sensor Network Platforms and Applications," *Proceedings of the International Microwave Symposium Digest*, pp. 4, 12-17 June 2005.

[118] S. Bandyopadhyay and E. Coyle, "An Energy Efficient Hierarchical Clustering Algorithm for Wireless Sensor Networks," *Proceedings of the 22nd Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM 2003)*, Vol. 3, pp. 1713-1723, March-April 2003.

[119] S. Das, C. Perkins and E. Royer, "Performance Comparison of Two On-Demand Routing Protocols for Ad Hoc Networks," *Proceedings of the 19th IEEE Annual Joint Conference of Computer and Communications Societies*, pp. 3–12, 2000.

[120] S. Lindsey and C.S. Raghavendra, "PEGASIS: Power Efficient Gathering in Sensor Information Systems," *Proceedings of the IEEE Aerospace Conference*, Vol. 3, pp. 1125-1130, March 2002.

[121] S. Lindsey, C.S. Raghavendra and K. Sivalingam, "Data Gathering in Sensor Networks using The Energy*Delay Metric," *Proceedings of the IPDPS Workshop on Issues in Wireless Networks and Mobile Computing*, pp. 924-935, April 2001.

[122] S. Pattem, B. Krishnamachari and R. Govindan, "The Impact of Spatial Correlation on Routing with Compression in Wireless Sensor Networks," *ACM Transactions on Sensor Networks (TOSN)*, Vol. 4, No. 4, April 2008.

[123] S. Paul, S. Nandi and I. Singh, "A Dynamic Balanced-Energy Sleep Scheduling Scheme in Heterogeneous Wireless Sensor Network," *Proceedings of the 16th IEEE International Conference on Networks (ICON 2008)*, pp. 1-6, December 2008.

[124] S. Srivastava, S. Tripathi, D. Sanghi and A.K. Chaturvedi, "A Code Allocation Protocol for Maximizing Throughput in CDMA based Ad Hoc Networks,"

Proceedings of the IEEE Wireless Communications and Networking Conference (WCNC), pp.1385-1390, March 2003.

[125] S.K. Gupta, K.S. Rao and V. Bhatnagar, "K-Means Clustering Algorithm for Categorical Attributes," *Proceedings of the First International Conference on Data Warehousing and Knowledge Discovery (DaWaK99)*, pp. 203-208, September 1999.

[126] T. Hara and S.K. Madria, "Data Replication for Improving Data Accessibility in Ad Hoc Networks," *IEEE Transactions on Mobile Computing*, Vol. 5, No. 11, pp.1515–1532, 2006.

[127] T. Hara, "Cooperative Caching by Mobile Clients in Push-Based Information Systems," *Proceedings of the 11th International Conference on Information and Knowledge Management (CIKM '02)*, pp. 186–193, 2002.

[128] T. Hara, "Replica Allocation Methods in Ad Hoc Networks with Data Update," *Kluwer's Mobile Networks and Applications*, Vol. 8, No. 4, pp. 343–354, 2003.

[129] T. He, J.A. Stankovic, C. Lu and T. Abdelzaher, "SPEED: A Stateless Protocol for Real-Time Communication in Sensor Networks," *Proceedings of the 23rd International Conference on Distributed Computing Systems*, pp. 46-55, May 2003.

[130] T. Pering, V. Raghunathan and R. Want, "Exploiting Radio Hierarchies for Power-Efficient Wireless Discovery and Connection Setup," *Proceedings of the 18th International Conference on VLSI Design*, pp. 774-779, 2005.

[131] T. Shivaprakash, C. Aravinda, A.P. Deepak, S. Kamal, H.L. Mahantesh, K.R. Venugopal and L.M. Patnaik, "Efficient Passive Clustering and Gateway Selection in MANETs," *Proceedings of the 7th International Workshop on Distributed Systems (IWDC 2005)*, pp. 548-553, 2005.

[132] T. Stathopoulos, M. Lukac, D. McIntire, J. Heidemann, D. Estrin and W.J. Kaiser, "End-to-End Routing for Dual-Radio Sensor Networks," *Proceedings of the 26th IEEE International Conference on Computer Communications*, pp. 2252-2260, May 2007.

[133] T.P. Sharma, R.C. Joshi and M. Misra, "Cooperative Caching for Homogeneous Wireless Sensor Networks," *International Journal of Communication Networks*

and Distributed Systems (IJCNDS), Vol. 2, No. 4, pp. 424-451, 2009.

[134] T.P. Sharma, R.C. Joshi and M. Misra, "Data Filtering and Dynamic Sensing for Continuous Monitoring in Wireless Sensor Networks," *International Journal of Autonomous and Adaptive Communications (IJAACSS)* (To appear)

[135] T.P. Sharma, R.C. Joshi and M. Misra, "Dual Radio Based Cooperative Caching for Wireless Sensor Networks," *Proceedings of the 16th IEEE International Conference on Networks (ICON 08)*, pp. 1-7, 12-14 December 2008.

[136] T.P. Sharma, R.C. Joshi and M. Misra, "GBDD: Grid Based Data Dissemination in Wireless Sensor Networks," *Proceedings of the 16th International Conference on Advanced Computing and Communications (ADCOM 08)*, pp. 234-240, 14-17 December 2008.

[137] T.P. Sharma, R.C. Joshi and M. Misra, "Tuning Data Reporting and Sensing for Continuous Monitoring in Wireless Sensor Networks," *Proceedings of the 27th IEEE International Performance Computing and Communication Conference (IPCCC 08)*, pp. 412-417, December 2008.

[138] T.P. Sharma, R.C. Joshi and M. Misra, "Undesired Data Suppression in Wireless Sensor Networks," *Proceedings of the 4th IEEE International Conference on Wireless Communication and Sensor Networks (WCSN 08)*, pp. 14-19, 27-29 December, 2008.

[139] U. Roedig, A. Barroso and C.J. Sreenan. "Determination of Aggregation Points in Wireless Sensor Networks," *Proceedings of the 30th International Conference EUROMICRO*, pp. 503 - 510, 2004.

[140] V. Erramilli, I. Matta and A. Bestavros, "On the Interaction Between Data Aggregation and Topology Control in Wireless Sensor Networks," *Proceedings of the 1st IEEE Communications Society Conference on Sensor and Ad Hoc Communications and Networks (SECON 2004)*, pp. 557-565, October 2004.

[141] V. Rodoplu and T.H. Ming, "Minimum Energy Mobile Wireless Networks," *IEEE Journal of Selected Areas in Communications,* Vol. 17, No. 8, pp. 1333–1344, (1999).

[142] V. Varadharajan, R. Shankaran and M. Hitchens, "Security for Cluster Based Ad Hoc Networks," *Computer Communications,* Vol. 27, No. 5, pp. 488-501, 2004.

[143] W. Heinzelman, J. Kulik and H. Balakrishnan, "Adaptive Protocols for Information Dissemination in Wireless Sensor Networks," *Proceedings of the 5th Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom_99)*, pp. 174–185, August 1999.

[144] W. Xu, K. Ma, W. Trappe and Y. Zhang, "Jamming Sensor Networks: Attack and Defense Strategies," *IEEE Networks Special Issue on Sensor Networks*, Vol.20, No. 3, pp. 41-47, May/June 2006.

[145] W. Xu, W. Trappe and Y. Zhang, "Defending Wireless Sensor Networks from Radio Interference through Channel Adaptation," *ACM Transactions on Sensor Networks (TOSN)*, Vol. 4, No. 4, August 2008.

[146] W. Ye, J. Heidemann and D. Estrin, "An Energy-Efficient MAC Protocol for Wireless Sensor Networks", *Proceedings of the IEEE INFOCOM*, pp. 1567-1576, June, 2002.

[147] W.B. Heinzelman, A.P. Chandrakasan and H. Balakrishnan, "An Application Specific Protocol Architecture for Wireless Microsensor Networks," *IEEE Transactions on Wireless Communications*, Vol. 1, No. 4, pp. 660–670, 2002.

[148] W.B. Heinzelman, A.P. Chandrakasan and H. Balakrishnan, "Application Specific Protocol Architecture for Wireless Microsensor Networks," *IEEE Transactions on Wireless Communications*, Vol. 1, No. 4, pp. 660-670, 2002.

[149] W.H.O Lau, M. Kumar and S. Venkatesh, "Cooperative Cache Architecture in Support of Caching Multimedia Objects in MANETs," *Proceedings of the ACM International Workshop on Wireless Mobile Multimedia*, pp. 56–63, 2002.

[150] www.ambient-systems.net. *Ambient Systems. Enschede, The Netherlands.*

[151] www.eng.yale.edu/enalab/xyz. *XYZ Sensor Node. Yale University.*

[152] www.sensinode.com. *Sensinode product catalog. Oulu, Finland.*

[153] www.sentilla.com. *Sentilla. Redwood City, CA, USA.*

[154] www.standards.ieee.org. *IEEE standard 802.15.4.*

[155] www.sunspotworld.com. *Project Sun SPOT. Santa Clara, CA, USA.*

[156] www.xbow.com. Crossbow Wireless Sensor Networks, Product Reference Guide 2007. San Jose, CA, USA.

[157] www.zigbee.org. *ZigBee Alliance*

[158] X. Wang, Y.M. Teo and J. Cao, "Message and Time Efficient Consensus Protocols for Synchronous Distributed Systems," *Journal of Parallel and Distributed Computing*, Vol.68, Issue 5, pp. 641-654, May 2008.

[159] Y. Agarwal, C. Schurgers and R. Gupta, "Dynamic Power Management using On Demand Paging for Networked Embedded Systems," *Proceedings of the Asia South Pacific design automation Conference (ASP-DAC 2005)*, pp. 755-759, 2005.

[160] Y. Hu, N. Yu and X. Jia, "Energy Efficient Real-Time Data Aggregation in Wireless Sensor Networks," *Proceedings of the ACM International Conference on Communications and Mobile Computing*, pp. 803-808, July 2006.

[161] Y. Jin, L. Wang, Y. Kim and X. Yang, "EEMC: An Energy-Efficient Multi-Level Clustering Algorithm for Large-Scale Wireless Sensor Networks," *Computer Networks*, Vol. 2, No. 3, pp. 542–562, 2008.

[162] Y. Xu, J. Heidemann and D. Estrin, "Geography-informed Energy Conservation for Ad-hoc Routing," *Proceedings of the annual ACM/IEEE International Conference on Mobile Commuting and Networking*, pp. 70–*84, 2001.*

[163] Y. Yao and J. Gehrke, "Query Processing for Sensor Networks," *Proceedings of the ACM 1st First Biennial Conference on Innovative Data Systems Research (CIDR 2003)*, January 2003.

[164] Y. Yao and J. Gehrke, "The Cougar Approach to In-Network Query Processing in Sensor Networks," *ACM SIGMOD Record*, Vol. 31, No. 3, pp. 9–18, September 2002.

[165] Y. Yu, D. Estrin and R. Govindan, "Geographical and Energy Aware Routing: A Recursive Data Dissemination Protocol for Wireless Sensor Networks," *UCLA Computer Science Department Technical Report, UCLA-CSD TR-01-0023*, May 2001.

[166] Y. Zhu, K. Sundaresan and R. Sivakumar, "Practical Limits on Achievable Energy Improvements and Useable Delay Tolerance in Correlation Aware Data Gathering in Wireless Sensor Networks," *Proceedings of the IEEE International Conference on Sensor and Ad Hoc Communications and Networks (SECON*

2005), pp. 328-329, September 2005.

[167] Z. Jiang, J. Ma, W. Lou and J. Wu, "An Information Model for Geographic Greedy Forwarding in Wireless Ad-Hoc Sensor Networks," *Proceedings of the 27th IEEE conference on Computer Communications (INFOCOM 2008)*, pp. 825-833, April 2008.

Author's Research Publications

International Journals

1. T.P. Sharma, R.C. Joshi, Manoj Misra, "Cooperative Caching for Homogeneous Wireless Sensor Networks", *in the special issue of International Journal of Communication Networks and Distributed Systems (IJCNDS),* Vol. 2, No. 4, pp. 424-451, 2009.
2. T.P. Sharma, R.C. Joshi, Manoj Misra, "Data Filtering and Dynamic Sensing for Continuous Monitoring in Wireless Sensor Networks," *special issues of International Journal of Autonomous and Adaptive Communications (IJAACS)* (To appear).
3. T.P. Sharma, R.C. Joshi, Manoj Misra, "GBDD: Grid Based Data Dissemination in Wireless Sensor Networks," *International Journal of Parallel, Emergent and Distributed Systems* (second review submitted).
4. T.P. Sharma, R.C. Joshi, Manoj Misra, "TWSW: Two-Way Sliding Window Data Filtering Protocol for Wireless Sensor Networks," *International Journal of Sensor Networks (IJSNet)* (second review submitted).
5. T.P. Sharma, R.C. Joshi, Manoj Misra, "Dual Radio Based Virtual Optimal Grid for Wireless Sensor Networks," *International Journal of Personal and Ubiquitous Computing* (Communicated).

International Conferences

6. T.P. Sharma, R.C. Joshi, Manoj Misra, "Tuning Data Reporting and Sensing for Continuous Monitoring in Wireless Sensor Networks," *in the proceedings of 27th IEEE International Performance Computing and Communications Conference,* Austin, Texas, pp. 412-417, IEEE xplore DOI: 10.1109/PCCC.2008.4745084, 7-9 December, 2008.*(Best paper award)*
7. T.P. Sharma, R.C. Joshi, Manoj Misra, "Dual Radio Based Cooperative Caching for Wireless Sensor Networks," *in the proceedings of 16th IEEE International Conference on Networking (ICON 2008),* New Delhi, IEEE xplore DOI: 10.1109/ICON.2008.4772565, 12-14 December 2008.

8. T.P. Sharma, R.C. Joshi, Manoj Misra, "GBDD: Grid Based Data Dissemination in Wireless Sensor Networks," *in the proceedings of ACS International Conference on Advanced Computing and Communication (ADCOM 08)*, Chennai, pp.234-240, IEEE xplore DOI: 10.1109/ADCOM.2008.4760454, 14-17 December 2008.

9. T.P. Sharma, R.C. Joshi, Manoj Misra, "Undesired Data Suppression in Wireless Sensor Networks," in the proceedings of *4th IEEE International Conference on Wireless Communication and Sensor Networks (WCSN-2008)*, pp. 14-19, 27-29 Allahbad, December 2008.

10. Nidhi Bansal, T.P. Sharma, Manoj Misra, R.C. Joshi, "FTEP: A Fault Tolerant Election Protocol for Multi-level Clustering in Homogeneous Wireless Sensor Networks," *in the proceedings of 16th IEEE International Conference on Networking (ICON 2008)*, IEE xplore DOI: 10.1109/ICON.2008.4772563, New Delhi, 12-14 December 2008.

11. Nidhi Bansal, T.P. Sharma, Manoj Misra, R.C. Joshi, "EEEP: An Energy Efficient Election Protocol for Multi-level Clustering in Homogeneous Wireless Sensor Networks," *in the proceedings of International Conference on Computer Networks and Security (ICCNS 2008)*, Pune, pp. 262-268, 26-28 September 2008.

12. T.P. Sharma, R.C. Joshi, Manoj Misra, "Medium Access Control Layer and Network Layer Issues in Wireless Sensor Networks," *in the proceedings of ACS International Conference on Global Software Development*, Coimbtore, 26-28 July 2007.

National Conferences

13. T.P. Sharma, R.C. Joshi, Manoj Misra. "Directed Diffusion Protocols for Specific Scenarios in Wireless Sensor Networks," *in the proceedings of Conference ETCC'07*, Hamirpur, pp. 245-248, July 2007.

14. T.P. Sharma, R.C. Joshi, Manoj Misra, "Wireless Sensor Networks: Research Challenges," *in the proceedings of Conference ECCS'06*, Patiala, pp. 361-165, February 2006.

Errata

Page No	Original	Revised
65, Section 3.2.4	$D_{i,j}$	D_y
69,72	Size of sensor field 500x500 m^2	Size of sensor field 150x150 m^2
169	Reference 21 and 22 are same	
173	Reference 60 has appeared in both conference as well as IEEE transactions	
175	Reference 79 and 80 are same	