Performance Analysis of various Modulation Schemes for achieving energy efficient communication over fading channel for Wireless Sensor Network.

By

PIARUL ISLAM

BANDHON DAS

Department of Electronics & communications Engineering
EAST WEST UNIVERSITY

Under the Supervision of

MUSTAFA MAHMUD HUSSAIN

Assistant Professor
Department of Electronics & communications Engineering
East West University
Plot-A/2 Jahurul Islam City, Aftabnagar Main road, Dhaka-1219,Bangladesh.

Declaration

It is hereby declared that, this thesis titled " Performance Analysis of various Modulation Schemes for achieving energy efficient communication over fading channel for Wireless Sensor Network" has been accepted in partial fulfillment for the requirement of the degree of Bachelor of Science in Electronic & Communication Engineering on April.

Supervisor:

____

MUSTAFA MAHMUD HUSSAIN

Assistant Professor

Department of Electronics & Communications Engineering

East West University

Students:

____

PIARUL ISLAM

ID: 2012-1-55-062

____

BANDHON DAS

ID: 2012-1-55-056

Acceptance

This to certify that thesis titled as" Performance Analysis of various Modulation Schemes for achieving energy efficient communication over fading channel for Wireless Sensor Network" submitted to the respected members of the board of the examination of the faculty of engineering for partial fulfillment of the requirement's for the degree of bachelor of science in Electronics and Telecommunication engineering by the following students and has been accepted as satisfactory.

Submitted By

Piarul Islam
ID: 2012-1-55-062

Bandhon Das
ID: 2012-1-55-056

…………………………………...
Mustafa Mahmud Hussain
Assistant Professor
Department of Electronics & communications Engineering
East West University

…………………………………..
Dr.M.Mofazzal Hossain
Chairperson & Professor
Department of Electronics & communications Engineering
East West University

Acknowledgements

First of all, we are grateful to almighty Allah for overcoming all the difficulties & bringing in the thesis into reality.

We would like to convey our heartiest gratitude to our supervisor Mustafa Mahmud Hussain, Assistant Professor, East West University. Without his Continuous help , guideline & inspiration , it was impossible to publish this thesis paper.

We are deeply grateful to , Dr. M. Mofazzal Hossain Head of the Department of Electronic & Communication Engineering for his kind co-operation.

We also would like to express our gratitude to Department of Electronics & Communications Engineering of East West University for giving the facilities to complete this work.

Abstract

Wireless sensor networks (WSNs) Holds the potential to improve the efficiency of many systems, for instance, in building automation and process control.

Unfortunately, the current technology does not offer guaranteed energy efficiency and reliability for closed-loop stability.

The main contribution of this thesis is to provide a modeling, analysis, and design framework for WSN protocols used in control applications. The protocols are designed to minimize the energy consumption of the network which is cost effective, while meeting reliability and delay requirements from the application layer.

In this paper, we have tried to find out the average life time of the batteries calculated for BPSK, QPSK and QAM transmission over different channel models as a function of link distance. Here each modulation is operated at its Optimal SNR. We have also found that, for long transmission distance BPSK,QPSK are optimal choices but as the transmission distance shortens the optimal modulation size grows to 16-QAM even to 64 QAM.

INDEX:

Chapter 1 : **Introduction**

List of Figures:

List of Tables

Chapter - 1

Introduction :

1.1 WSN Overview:

Recently, technological advances in the design of processors, memory and radio
communications have propelled an active interest in the area of distributed sensor networking, in which a number of independent, self-sustainable nodes collaborate
to perform information gathering and processing in real time.
Networks of such devices are commonly referred to as Wireless Sensor Networks
(WSNs), which are envisioned as a bridge between the modern broadband packet
data networks and the physical world. WSNs have made possible real-time data
aggregation and analysis on an unprecedented scale.

Wireless communication is the transfer of information between two or more points that are not connected by electrical conductors. Most of the wireless communication technology uses radio waves in order to transfer information between the points which are known as nodes. One application domain of wireless communication is wireless sensor networks. WSN is a distributed system, containing resource or constrained nodes that work in an ad hoc manner using multi-hope communication.

WSNs and Internet are integrated as a new application area called Internet of Things (IoT), covering almost every area in current daily life. IoT encourages several novel and existing applications such as environment monitoring, infrastructure management, public safety, medical and health care, home and office security, transportation, and military applications. Figure 1-1 shows the complexity of wireless sensor networks , which translate sensing and identification activities into services using WSNs with WSN middleware and access networking.

It can use: (i) different communication platforms such as Wi-Fi, wireless LAN, 3G and 4G; (ii) different devices which are based on different processors such as various types of PDA, smart phones and laptops and (iii) all these platforms and devices being built on different architectures such as centralized, distributed or peer-to-peer.

wireless sensor networks continue to grow because of their application scenarios and cost effectiveness. A major benefit of these systems is that they perform in network processing to reduce large streams of raw data into useful aggregated information . Protecting information is critical. The traditional computer network security goal is to deliver the message to the end user in a reliable way. The leading traffic pattern in the conventional computer network is end to end communication. The message content is not important beyond the necessary header. In this process the message authenticity, integrity and confidentiality are usually achieved by an end to end security mechanism such as Secure Socket Layer (SSL).

The use of wireless communication technology in WSNs introduces more challenges compared to that of fixed wired networks.

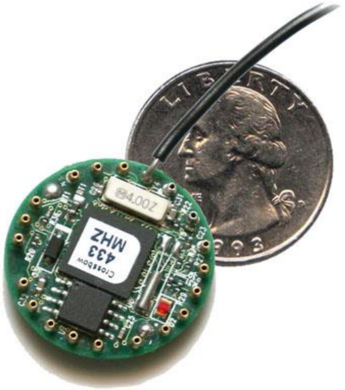


Figure 1: **Wireless sensor network device designed to be the approximate size of a quarter. Future devices will continue to be smaller, cheaper and longer lasting.**

1.2: Thesis Motivation

Wireless networks play a crucial role in the communication systems nowadays. Wireless networks are being increasingly used in the communication among devices of the most varied types and sizes. User mobility, affordable-ity, flexibility and ease of use are few of many reasons for making them very appealing to new applications and more users everyday. In this work, we consider only wireless networks capable of operating without the support of any fixed infrastructure. We also consider the general case of multi-hop networks. More precisely, we will consider wireless ad hoc networks as well as wireless sensor networks. The diversity of the applications supported by wireless ad hoc and sensor networks explain the success of this type of network. These applications concern as various domains as environmental monitoring, wildlife protection, emergency rescue, home monitoring, target tracking, exploration mission in hostile environments, etc. However, the most critical requirement for adopting such networks is energy efficiency. Indeed, some nodes are battery operated and battery replacement can be difficult, expensive or even impossible. The goal of communication protocol designers is then to maximize the lifetime of such networks. In this work various studied about the improving the lifetime of sensor node. The energy minimization is achieved by optimizing the modulation size with a transmission distance.

1.3 Problem Statement

In wireless sensor networks, nodes working only with battery power will die after

battery exhaustion. This means that the network has a limited lifetime. One of the main challenges facing the network designers in wireless sensor networks is to maximize the network lifetime. Our work is centered on energy efficient techniques that will prolong network lifetime. Several classes of energy efficient techniques exist. Among them In this paper, we have tried to find out the average life time of the batteries calculated for BPSK, QPSK and QAM transmission over different channel models as a function of link distance. Here each modulation is operated at its Optimal SNR.

1.4: Problem Formulation:

The goal of this thesis is to model, analyze, and design WSN protocols. As part of this work, we will:

1. Model the important performance indicators, such as reliability, delay, energy consumption, using mathematical tools, and

2. Analyze the resulting performance of the protocol by means of the experiments and simulations.

By using the derive protocol model, we use a general constrained optimization problem for the designs. Our objective is to minimize the total energy consumption of each node or all nodes of the network, denoted by *E*tot(u) where u is a vector of decision variables. The application requirements impose constraints on the reliability and packet delay. Hence, the optimization problem is

$$\min_{u} \quad E\text{tot}(u) \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (1.1a)$$

$$\text{s.t. } u \in R \cap D \cap F \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (1.1b)$$

The decision variables **u** are the protocol parameters of the physical layer (PHY),MAC, and routing layer. *R* and *D* are the feasible sets for the protocol parameters that meet the reliability and delay constraints, respectively. In addition, the feasible set *F* is due to physical layer properties of the hardware platform or limitations of the protocol standards. The derivation of analytical expressions of the energy consumption of the network, as well as reliability and delay for the packet delivery, is essential for the solution to the optimization problem. Therefore, the analytical modeling is a critical step to the protocol design in this thesis. Problem (1.1) is a mixed integer-real optimization problem, because **u** may take on both real and integer values. We model the components of Problem (1.1) and we derive a strategy to obtain its optimal solution, **u**⍰. As we will see later, the system complexity prevents us from deriving exact expressions for reliability, delay, and energy consumption. Approximations will be used to get tractable analytical models. Note that this constrained optimization problem can be local, in the sense that it is solved at a local node of the network using locally measurable information, or global, in the sense that includes information from the overall network and is solved centrally. Next, we give an example of a local optimization and an example of a global optimization, which are used in the thesis to design protocols.

Example 1.

In Chapter 5, a global optimization problem is introduced to optimize the wakeup rate and the number of hops in the network. The cross-layer protocol solution, called Breath, is designed for industrial control applications where source nodes attached to the plant must transmit information via multi-hop routing to a sink.

The protocol is based on randomized routing, MAC, and duty-cycling to minimize the energy consumption, while meeting reliability and packet delay constraints. The optimization problem is

$$\mathbf{Min} \quad \boldsymbol{E}\mathbf{tot(u)\ (1.3a)}$$
$$\mathbf{u}$$
$$\mathbf{s.t.}\ \boldsymbol{R} = \boldsymbol{\{}\mathbf{u/}\boldsymbol{R}\mathbf{(u)} \geq \boldsymbol{R}\mathbf{min}\boldsymbol{\}}\ \boldsymbol{,}\ \mathbf{(1.3b)}$$
$$\boldsymbol{D} = \boldsymbol{\{}\mathbf{u/\ Pr[}\boldsymbol{D}\mathbf{(u)} \leq \boldsymbol{D}\mathbf{max]} \geq \boldsymbol{\Omega}\boldsymbol{\}}\ \boldsymbol{,}\ \mathbf{(1.3c)}$$

where *E*tot is the energy consumption, and *R* and *D* are the feasible sets for the protocol parameters that meet the reliability and delay constraints of the entire network, respectively. The decision variables are the wake-up rate and the number of hops, which are achieved by collaboration between the nodes in the network. The optimization problem is based on an analytical model for energy consumption, reliability, and delay of the network.

1.4: Thesis Organization:

Chapter 1 , here we discuss about the w and our thesis motivation, challenges and problems and how the whole thesis organized.

Chapter 2 also introduces the main characteristics, architecture, and existing node platform and application scenarios, which motivate the work performed in the thesis. Here we also discuss about the lifetime, coverage, response and hardware capabilities like energy, battery technology etc.

Chapter 3 we talk about energy efficiency ,energy consumption model , data transfer rate , why energy consumption occurs in the network.

Chapter 4 , we discuss about the basic of PSK and QAM modulation characteristics ,application its transceiver model and AWGN and RAYLIEGH fading channel.

Chapter 5 , here we show the optimal SNR and transceiver life time equation

Chapter 6 , here we discuss about the result of simulation and analysis the result for make decision.

Chapter 7: concludes this dissertation and provides directions for further researches.

Chapter – 2

Wireless Sensor Networks

The concept of wireless sensor networks is based on a simple equation:

Sensing + CPU + Radio = Thousands of potential applications_As soon as people understand the capabilities of a wireless sensor network, hundreds of applications spring to mind. It seems like a straightforward combination of modern technology. However, actually combining sensors, radios, and CPU's into an effective_wireless sensor network requires a detailed understanding of the both capabilities and limitations of each of the underlying hardware components, as well as a detailed understanding of modern networking technologies and distributed systems theory. Each individual node must be designed to provide the set of primitives necessary to synthesize the interconnected web that will emerge as they are deployed, while meeting strict requirements of size, cost and power consumption. A core challenge is to map the overall system requirements down to individual device capabilities, requirements and actions. To make the wireless sensor network vision a reality, an architecture must be developed that synthesizes the envisioned applications out of the underlying hardware capabilities.

To develop this system architecture we work from the high level application requirements down through the low-level hardware requirements. In this process we first attempt to understand the set of target applications. To limit the number of applications that we must consider, we focus on a set of application classes that we believe are representative of a large fraction of the potential usage scenarios. We use this set ofapplication classes to explore the system-level requirements that are placed on the overall architecture. From these system-level requirements we can then drill down into the individual node-level requirements. Additionally, we must provide a detailed background into the capabilities of modern hardware. After we present the raw hardware capabilities, we present a basic wireless sensor node. The Rene node represents a first cut at a system architecture, and is used forcomparison against the system architectures .

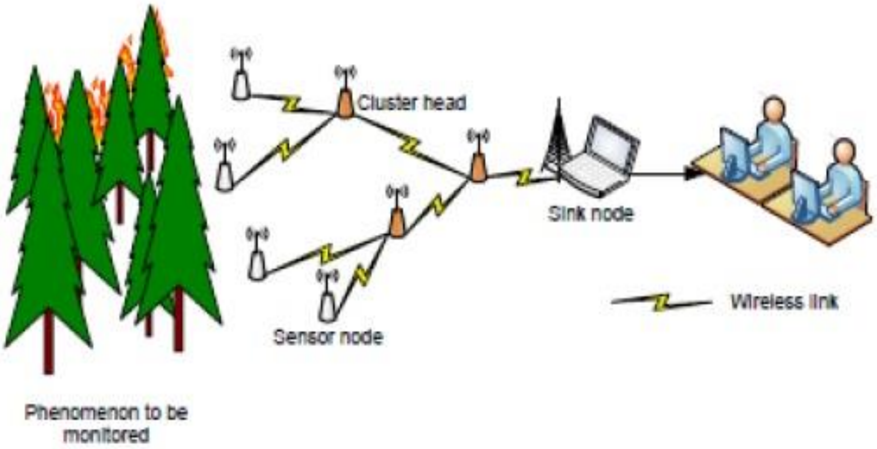


Figure 2: A Typical wireless sensor network

2.1.1 DIFFERENCE BETWWEEN AD HOC AND WSN

Wireless ad hoc and sensor networks have in common some characteristics that have to be taken into account by energy efficient techniques:

-Lack of pre-configuration: A wireless ad hoc and sensor network is a collection of wireless nodes that can dynamically self-organize into an arbitrary and temporary topology to form a network without using any pre-existing infrastructure.

Wireless communication: which has the following properties:

-Radio interferences: Indeed, when a node N1 is transmitting to a neighbor node N2, no other node in the transmission range of N1 can receive another frame. Similarly, no other node in the transmission range of N2 can send another frame.

-Radio link versatility: as the propagation conditions change very frequently, the quality of a radio link varies strongly in the time.

– Limited bandwidth: the wireless bandwidth has a capacity much smaller than a wired
one due to the shared nature of the wireless channel and interferences

2.1.2 CHARACTERISTICS OF WSN:

Wireless sensor networks pose unique security challenges. The traditional security techniques used in traditional networks cannot be efficiently applied to WSNs directly to deal with attacks, because WSNs have the following characteristics:

□□The sensor networks should be economically viable as sensor devices arelimited in their energy, computation, and communication capabilities.

□□Unlike traditional cases wireless sensor nodes are often deployed inaccessible areas, which presents the additional risk of physical attack.

□□Wireless sensor networks normally have open media within thedeployment environment, which increases challenges to the security.

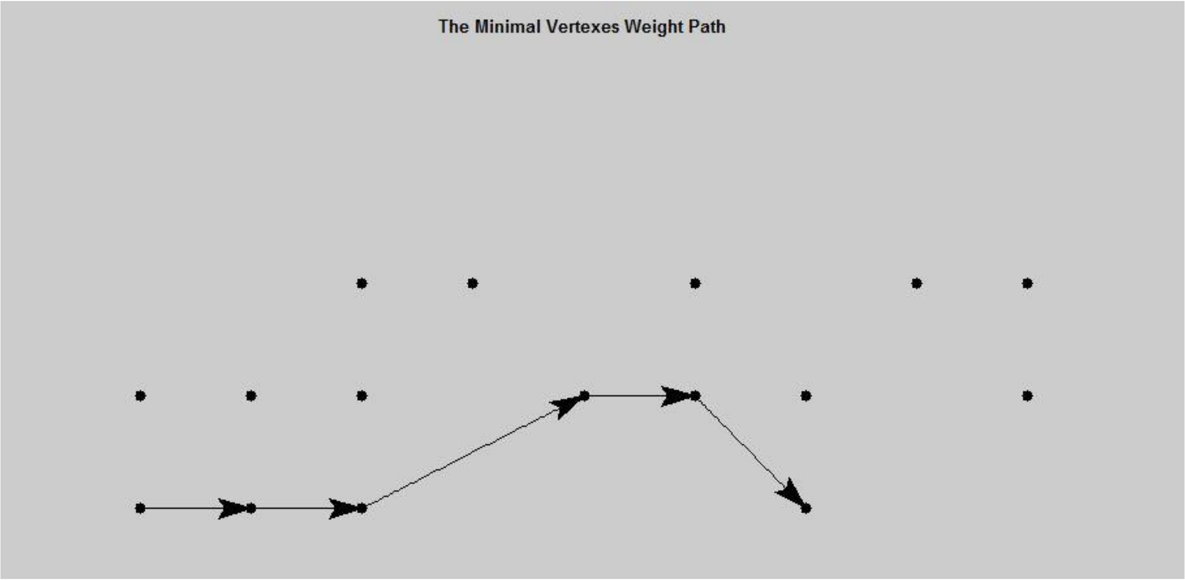


Figure 3 : Minimum Vertexes Weight Path of WSN

2.1.3 APPLICATIONS OF WSN

WSNs consist of spatially distributed autonomous devices to cooperatively monitor real world physical or environmental conditions, such as temperature, sound, vibration, pressure, motion, pollution and location. This technology is also widely used by military applications, such as battlefield surveillance, transportation monitoring, and sensing of nuclear, biological and chemical agents. Recently, this technology has developed and been widely used in daily life as WSNs are low cost, low power, rapid deployment, have self- organizing capability and cooperative data processing, including applications forhabitat monitoring, intelligent agriculture and home automation.

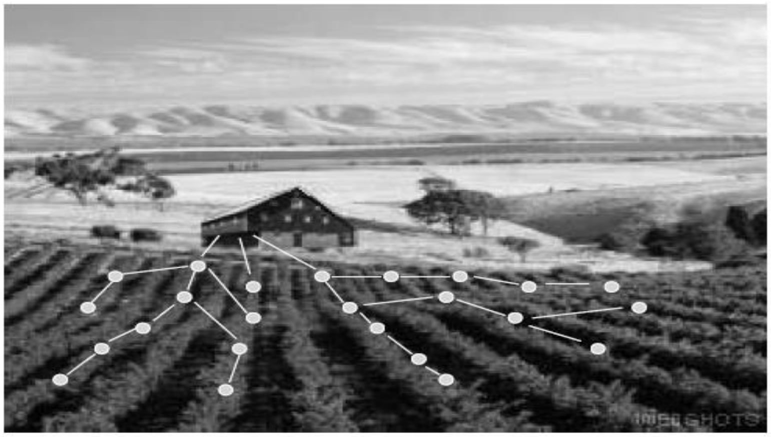


Figure 4: **Sensors detect temperature, light levels and soil moisture at hundreds of points across a field**

2.1.4 BASIC STRUCTURE

The major components of a normal WSN sensor node are a microcontroller, memory, transceiver, power source and one or more sensors to detect the physical phenomena. The structure of the sensor node is generally divided into four major parts: sensing unit, processing unit, communication unit and power unit. A sensor node sends the measurement of the physical phenomenon to the sink which has bigger memory and processing power. Depending on the application scenario, sometimes extra hardware is added in the sensor nodes and a deployment strategy is devised. Normally, in applications for WSNs the environment is unpredictable such as hostile, with remote harsh fields or disaster areas, sometimes called toxic environments.

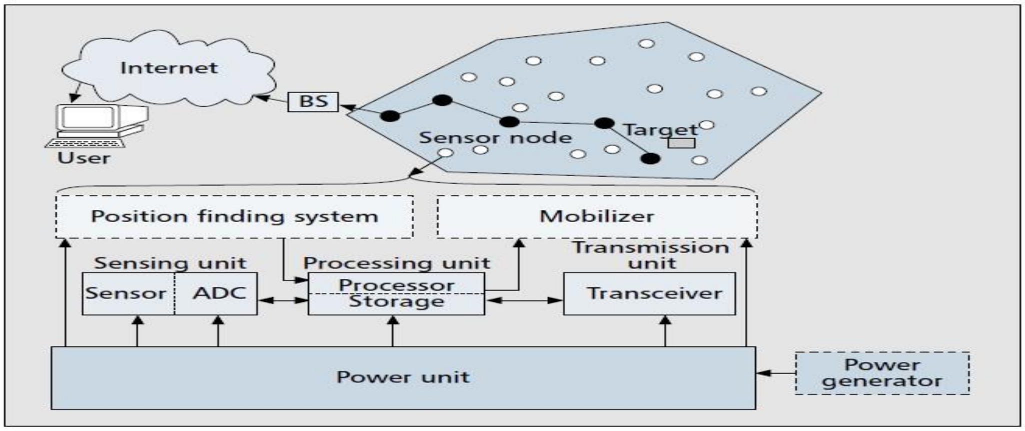


Fig.5.The components of a sensor node and WSN

2.1.5 PROTOCOL STACK OF WIRELESS SENSOR NETWORK

Power unit: The task of the power unit is to provide the energy to the sensor node for monitoring the environment at a low cost and less time. The life of the sensor depends on the battery or power generator which is connected to the power unit.
Power unit is required for an efficient use of the battery.
When the knowledge about the structure of a sensor node is acquired, it is necessary to further check and understand the communication architecture of WSNs. The communication architecture of a WSN is slightly different from the conventional computer communication and

computer network. The major entities that build up the communication architecture are :

□□The sensor node objectives are to make discrete, local measurements of phenomena surrounding these sensors, forming a wireless sensor network by communicating over a multi-hop wireless medium, and collect data and rout data back to the user via a sink or a base station.
□□The sink (Base Station) communicates with the user via a suitable communication method such as internet, satellite, Wimax, WiFi, 3G or 4G. It is located near the sensor field or well-equipped nodes of the sensor network. Collected data from the sensor field routed back to the sink by a hop to hop infrastructure.

□□Phenomenon expressed by related physical parameters, which is an entity of interest to the user to collect measurements about specific phenomenon. This phenomenon sensed and analyzed by the sensor nodes of a WSN.

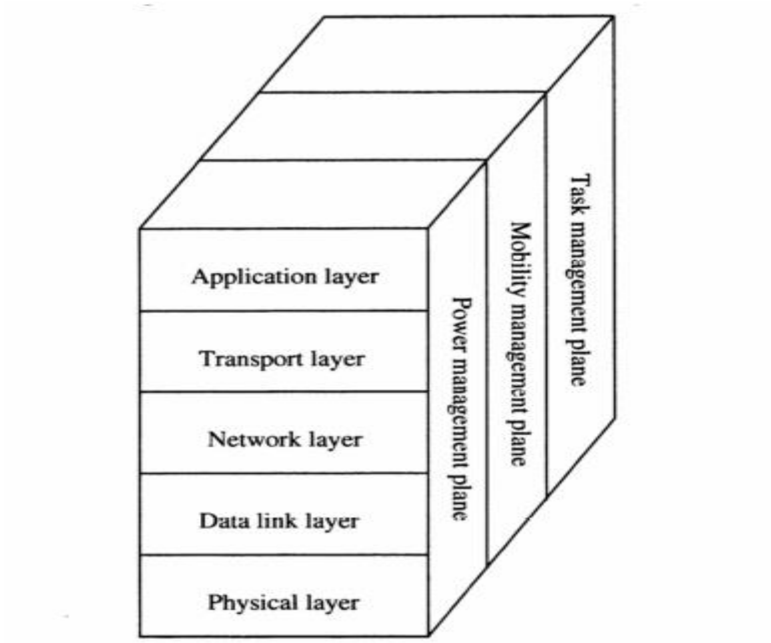


Figure 6 : Protocol stack of WSN

The physical layer is responsible for frequency selection such as carrier frequency generation which corresponds to checking RFID data list to make sure the task, signal detection, modulation, and data encryption are running well. The data link layer is concerned with the media access control (MAC) protocol. Since the wireless channel is normally affected by the noise and sensor nodes may be changing the location, the MAC protocol at the data link layer has to be power-aware and should have the capability of minimizing the collisions . The
network layer manages the routing data supplied by the transport layer or between the nodes. Whereas the transport layer is able to maintain the data flow if the WSN's application requires that. Various types of application can be implemented in the application layer depending on the physical environmental sensing. Orthogonal to the five layers, Akyildiz et al. defined three management plans named power, mobility and task management. Theseplans are responsible for monitoring the power, movement and task distribution among the sensor nodes. These management plans help the sensor nodes to coordinate sensor tasks and minimize the overall power consumption.

2.1.6 Protocols of WSNs

WSNs are designed to carry out various tasks which are underpinned by several protocols. This section are going to discuss some major related protocols for WSNs. Routing protocols of WSNs are inspired by ad hoc networking for some similarities in their characteristics [41]. Moreover, WSNs have some specific properties such as coverage cast traffic profile, strong energy constrain, densely deployed high number of nodes .Thus, it is necessary to take special care for WSNs. There are different ways to classify the sensor networks routing protocols. According to Ochirkhand , the classification of routing protocol can be divided into four categories: Flooding based routing, Probabilistic routing, Location based routing and Hierarchical routing, as shown in the Figure 7

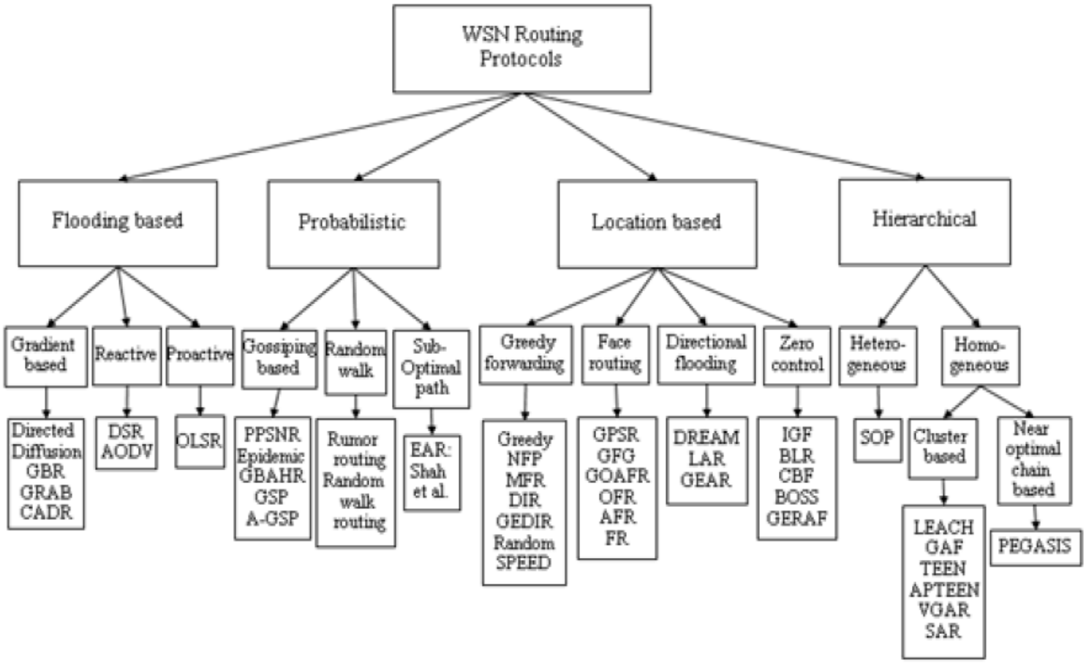


Figure 7 : Routing protocols of WSNs

Flooding based routing is a static algorithm which uses flooding mechanism to discover routs. In flooding based protocol every incoming packet is sent out on every outgoing line except the one it arrived on .Flooding based generates infinite number of duplicate packets unless some measures are taken to damp the process. Probabilistic routing chooses the next hope using a dynamically assigned probability or random choice making their behaviour non-deterministic
. The location based routing protocols uses geographical location information to guide routing discovery and maintenance as well as data forwarding, enabling directional transmission of the information and avoiding information flooding in the entire network [. Each node needs to know its destination, its own location and the location of the neighbor. Hierarchical routing is based on hierarchy among the nodes when a larger amount of resources is necessary to take care or a routing table becomes enormous and makes routing impossible.

The idea of hierarchical routing suggests that routers should be divided into regions, with each router knowing all the details about how to route packets within its own region, but knowing nothing about the internal structure of other regions. The list of a few popular routing protocols for wireless sensor networks below .

- Direct diffusion
- GBR (Gradient Based Routing)
- AODV (Ad hoc On-Demand Distance Vector)
- GPSR (Greedy Perimeter Stateless Routing)
- LEACH (Low Energy Adaptive Clustering Hierarchy)

2.2 Sensor network application classes

The three application classes we have selected are: environmental data collection, security monitoring, and sensor node tracking. We believe that the majority of wireless sensor network deployments will fall into one of these class templates.

2.2.1 Environmental Data Collection

A canonical environmental data collection application is one where a research scientist wants to collect several sensor readings from a set of points in an environment over a period of time in order to detect trends and interdependencies. This scientist would want to collect data from hundreds of points spread throughout the area and then analyze the data offline. The scientist would be interested in collecting data over several months or years in order to look for long-term and seasonal trends. For the data to be meaningful it would have to be collected at regular intervals and the nodes would remain at known locations. At the network level, the environmental data collection application is characterized by having a large number of nodes continually sensing and transmitting data back to a set of base stations that store the data using traditional methods. These networks generally require very low data rates and extremely long lifetimes. In typical usage scenario, the nodes will be evenly distributed over an outdoor environment. This distance between adjacent nodes will be minimal yet the distance across the entire network will be significant. After deployment, the nodes must first discover the topology of the network and estimate optimal routing strategies . The routing strategy can then be used to route data to a central collection points. In environmental monitoring applications, it is not essential that the nodes develop the optimal routing strategies on their own. Instead, it may be possible to calculate the optimal routing topology outside of the network and then communicate the necessary information to the nodes as required. This is possible because the physical topology of the network is relatively constant. While the time variant nature of RF communication may cause connectivity between two nodes to be intermittent, the overall topology of the network will be relatively stable. Environmental data collection applications typically use tree-based routing topologies where each routing tree is rooted at high-capability nodes that sink data. Data is periodically transmitted from child node to parent node up the tree-structure until it reaches the sink. With tree-based data collection each node is responsible for forwarding the data of all its descendants. Nodes with a large number of descendants transmit significantly more data than leaf nodes. These nodes can quickly become energy bottlenecks .Once the network is configured, each node periodically samples its sensors and transmits its data up the routing tree and back to the base station. For many scenarios, the interval between these transmissions can be on the order of minutes. Typical reporting periods are expected to be between 1 and 15 minutes; while it is possible for networks to have significantly higher reporting rates. The typical environment parameters being monitored, such as temperature, light intensity, and humidity, do not change quickly enough to require higher reporting rates. In addition to large sample intervals, environmental monitoring applications do not have strict latency requirements. Data samples can be delayed inside the network for moderate periods of time without significantly affecting application performance. In general the data is collected for future analysis, not for real-time operation. In order to meet lifetime requirements, each communication event must be precisely scheduled. The senor nodes will remain dormant a majority of the time; they will only wake to transmit or receive data. If the precise schedule is not met, the communication events will fail. As the network ages, it is expected that nodes will fail over time. Periodically the network will have to reconfigure to handle node/link failure or to redistribute network load. Additionally, as the researchers learn more about the environment they study, they may want to go in and insert additional sensing points. In both cases, the reconfigurations are relatively infrequent and will not represent a significant amount of the overall system energy usage. The most important characteristics of the environmental monitoring requirements are long lifetime, precise synchronization, low data rates and relatively static topologies. Additionally it is not essential that the data be transmitted in real-time back to the central collection point. The data transmissions can be delayed inside the network as necessary inorder to improve network efficiency.

2.2.2 Security Monitoring

Our second class of sensor network application is security monitoring. Security monitoring networks are composed of nodes that are placed at fixed locations throughout an environment that continually monitor one or more sensors to detect an anomaly. A key difference between security monitoring and environmental monitoring is that security networks are not actually collecting any data. This has a significant impact on the optimal network architecture. Each node has to frequently check the status of its sensors but it only has to transmit a data report when there is a security violation. The immediate and reliable communication of alarm messages is the primary system requirement. These are "report by exception" networks. Additionally, it is essential that it is confirmed that each node is still present and functioning. If a node were to be disabled or fail, it would represent a security violation that should be reported. For security monitoring applications, the network must be configured so that nodes are responsible for confirming the status of each other. One approach is to have each node be assigned to peer that will report if a node is not functioning. The optimal topology of a security monitoring network will look quite different from that of a data collection network. In a collection tree, each node must transmit the data of all of its decedents. Because of this, it is optimal to have a short, wide tree. In contrast, with a security network the optimal configuration would be to have a linear topology that forms a Hamiltonian cycle of the network. The power consumption of each node is only proportional to the number of children it has. In a linear network, each node would have only one child. This would evenly distribute the energy consumption of the network.

The accepted norm for security systems today is that each sensor should be checked approximately once per hour. Combined with the ability to evenly distribute the load of checking nodes, the energy cost of performing this check becomes minimal. A majority of the energy consumption in a security network is spent on meeting the strict latency requirements associated with the signaling the alarm when a security violation occurs. Once detected, a security violation must be communicated to the base station immediately. The latency of the data communication across the network to the base station has a critical impact on application performance. Users demand that alarm situations be reported within seconds of detection. This means that network nodes must be able to respond quickly to requests from their neighbors to forward data. In security networks reducing the latency of an alarm transmission is significantly more important than reducing the energy cost of the transmissions. This is because alarm events are expected to be rare. In a fire security system alarms would almost never be signaled. In the event that one does occur a significant amount of energy could be dedicated to the transmission. Reducing the transmission latency leads to higher energy consumption because routing nodes must monitor the radio channel more frequently. In security networks, a vast majority of the energy will be spend on confirming the functionality of neighboring nodes and in being prepared to instantly forward alarm announcements. Actual data transmission will consume a small fraction of the network energy.

2.2.3 Node tracking scenarios

A third usage scenario commonly discussed for sensor networks is the tracking of a tagged object through a region of space monitored by a sensor network. There are many situations where one would like to track the location of valuable assets or personnel. Current inventory control systems attempt to track objects by recording the last checkpoint that an object passed through. However, with these systems it is not possible to determine the current location of an object. For example, UPS tracks every shipment by scanning it with a barcode whenever it passes through a routing center. The system breaks down when objects do not flow from checkpoint to checkpoint. In typical work environments it is impractical to expect objects to be continually passed through checkpoints. With wireless sensor networks, objects can be tracked by simply tagging them with a small sensor node. The sensor node will be tracked as it moves through a field of sensor nodes that are deployed in the environment at known locations. Instead of sensing environmental data, these nodes will be deployed to sense the RF messages of the nodes attached to various objects. The nodes can be used as active tags that announce the presence of a device. A database can be used to record the location of tracked objects relative to the set of nodes at known locations. With this system, it becomes possible to ask where an object is currently, not simply where it was last scanned .Unlike sensing or security networks, node tracking applications will continually have topology changes as nodes move through the network. While the connectivity between the nodes at fixed locations will remain relatively stable, the connectivity to mobile nodes will be continually changing. Additionally the set of nodes being tracked will continually change as objects enter and leave the system. It is essential that the network be able to efficiently detect the presence of new nodes that enter the network.

2.2.4 Hybrid networks

In general, complete application scenarios contain aspects of all three categories. For example, in a network designed to track vehicles that pass through it, the network may switch between being an alarm monitoring network and a data collection network. During the long periods of inactivity when no vehicles are present, the network will simply perform an alarm monitoring function. Each node will monitor its sensors waiting to detect a vehicle. Once an alarm event is detected, all or part of the network, will switch into a data collection network and periodically report sensor readings up to a base station that track the vehicles progress. Because of this multi-modal network behavior, it is important to develop a single architecture that and handle all three of these application scenarios.

2.3 System Evaluation Metrics

Now that we have established the set of application scenarios that we are addressing, we explore the evaluation metrics that will be used to evaluate a wireless sensor network. To do this we keep in mind the high-level objectives of the network deployment, the intended usage of the network, and the key advantages of wireless sensor networks over existing technologies. The key evaluation metrics for wireless sensor networks are lifetime, coverage, cost and ease of deployment, response time, temporal accuracy, security, and effective sample rate. Their importance is discussed below. One result is that many of these evaluation metrics are interrelated. Often it may be necessary to decrease performance in one metric, such as sample rate, in order to increase another, such as lifetime. Taken together, this set of metrics form a multidimensional space that can be used to describe the capabilities of a wireless sensor network. The capabilities of a platform are represented by a volume in this multidimensional space that contains all of the valid operating points.In turn, a specific application deployment is represented by a single point. A system platform can successfully perform the application if and only if the application requirements point lies inside the capability hyperspace. One goal of this chapter is to present an understanding of the tradeoffs that link each axis of this space and an understanding of current capabilities. The architectural improvements and optimizations we present in later chapters are then motivated by increasing the ability to deliver these capabilities and increasing the volume of the capability hypercube.

2.3.1 Lifetime

Critical to any wireless sensor network deployment is the expected lifetime. The goal of both the environmental monitoring and security application scenarios is to have nodes placed out in the field, unattended, for months or years. The primary limiting factor for the lifetime of a sensor network is the energy supply. Each node must be designed to manage its local supply of energy in order to maximize total network lifetime. In many deployments it is not the average node lifetime that is important, but rather the minimum node lifetime. In the case of wireless security systems, every node must last for multiple years. A single node failure would create a vulnerability in the security systems. In some situations it may be possible to exploit external power, perhaps by tapping into building power with some or all nodes. However, one of the major benefits to wireless systems is the ease of installation. Requiring power to be supplied externally to all nodes largely negates this advantage. A compromise is to have a handful of

special nodes that are wired into the building's power infrastructure. In most application scenarios, a majority of the nodes will have to be selfpowered. They will either have to contain enough stored energy to last for years, or they will have to be able to scavenge energy from the environment through devices, such as solar cells or piezoelectric generators .Both of these options demand that that the average energy consumption of the nodes be as low as possible. The most significant factor in determining lifetime of a given energy supply is
radio power consumption. In a wireless sensor node the radio consumes a vast majority of the system energy. This power consumption can be reduced through decreasing the transmission output power or through decreasing the radio duty cycle. Both of these alternatives involve sacrificing other system metrics.

2.3.2 Coverage

Next to lifetime, coverage is the primary evaluation metric for a wireless network. It is always advantageous to have the ability to deploy a network over a larger physical area. This can significantly increase a system's value to the end user. It is important to
keep in mind that the coverage of the network is not equal to the range of the wireless communication links being used. Multi-hop communication techniques can extend the coverage of the network well beyond the range of the radio technology alone. In theory
they have the ability to extend network range indefinitely. However, for a given transmission range, multi-hop networking protocols increase the power consumption of the nodes, which may decrease the network lifetime. Additionally, they require a minimal node density, which may increase the deployment cost. Tied to range is a network's ability to scale to a large number of nodes. Scalability is a key component of the wireless sensor network value proposition. A user can deploy a small trial network at first and then can continually add sense points to collect more and different information. A user must be confident that the network technology being used is capable of scaling to meet his eventual need. Increasing the number of nodes in the system will impact either the lifetime or effective sample rate. More sensing points

will cause more data to be transmitted which will increase the power consumption of the network. This can be offset by sampling less often.

2.3.3 Cost and ease of deployment

A key advantage of wireless sensor networks is their ease of deployment. Biologists and construction workers installing networks cannot be expected to understand the underlying networking and communication mechanisms at work inside the wireless network. For system deployments to be successful, the wireless sensor network must configure itself. It must be possible for nodes to be placed throughout the environment by an untrained person and have the system simply work. Ideally, the system would automatically configure itself for any possible physical node placement. However, real systems must place constraints on actual node placements – it is not possible to have nodes with infinite range. The wireless sensor network must be capable of providing feedback as to when these constraints are violated. The network should be able to assess quality of the network deployment and indicate any potential problems. This translates to requiring that each device be capable of performing link discovery and determining link quality. In addition to an initial configuration phase, the system must also adapt to changing environmental conditions. Throughout the lifetime of a deployment, nodes may be relocated or large physical objects may be placed so that they interfere with the communication between two nodes. The network should be able to automatically reconfigure on demand in order to tolerate these occurrences.

The initial deployment and configuration is only the first step in the network lifecycle. In the long term, the total cost of ownership for a system may have more to do with the maintenance cost than the initial deployment cost. The security application scenario in particular requires that the system be extremely robust. In addition to extensive hardware and software testing prior to deployment, the sensor system must be constructed so that it is capable of performing continual selfmaintenance. When necessary, it should also be able to generate requests when external maintenance is required. In a real deployment, a fraction of the total energy budget must be dedicated to system maintenance and verification. The generation of diagnostic and reconfiguration traffic reduces the network lifetime. It can also decrease the effective sample rate.

2.3.4 Response Time

Particularly in our alarm application scenario, system response time is a critical performance metric. An alarm must be signaled immediately when an intrusion is detected. Despite low power operation, nodes must be capable of having immediate, high-priority messages communicated across the network as quickly as possible. While these events will be infrequent, they may occur at any time without notice. Response
time is also critical when environmental monitoring is used to control factory machines and equipment. Many users envision wireless sensor networks as useful tools for industrial process control. These systems would only be practical if response time guarantees could be met.
The ability to have low response time conflicts with many of the techniques used to increase network lifetime. Network lifetime can be increased by having nodes only operate their radios for brief periods of time. If a node only turns on its radio once per minute to transmit and receive data, it would be impossible to meet the application requirements for response time of a security system. Response time can be improved by including nodes that are powered all the time.
These nodes can listen for the alarm messages and forward them down a routing backbone when necessary. This, however, reduces the ease of deployment for the system.

2.3.5 Temporal Accuracy

In environmental and tracking applications, samples from multiple nodes must be cross-correlated in time in order to determine the nature of phenomenon being measured. The necessary accuracy of this correlation mechanism will depend on the rate of propagation of the phenomenon being measured. In the case of determining the average temperature of a building, samples must only be correlated to within seconds. However, to determine how a building reacts to a seismic event, millisecond accuracy is required. To achieve temporal accuracy, a network must be capable of constructing and maintaining a global time base that can be used to chronologically order samples and events. In a distributed system, energy must be expended to maintain this distributed clock. Time synchronization information must be continually communicated between nodes. The frequency of the synchronization messages is dependent on the desired accuracy of the time clock. The bottom line is maintenance of a distributed time base requires both power and bandwidth.

2.3.6 Security

Despite the seemingly harmless nature of simple temperature and light information from an environmental monitoring application, keeping this information secure can be extremely important. Significant patterns of building use and activity can be easily extracted from a trace of temperature and light activity in an office building. In the wrong hands, this information can be exploited to plan a strategic or physical attack on a company. Wireless sensor networks must be capable of keeping the information they are collecting private from eaves dropping. As we consider security oriented applications, data security becomes even more significant. Not only must the system maintain privacy, it must also be able to authenticate data communication. It should not be possible to introduce a false alarm message or to replay an old alarm message as a current one. A combination of privacy and authentication is required to address the needs of all three scenarios. Additionally, It should not be possible to prevent proper operation by interfering with transmitted signals. Use of encryption and cryptographic authentication costs both power and network bandwidth . Extra computation must be performed to encrypt and decrypt data and extra authentication bits must be transmitted with each packet. This impacts_application performance by decreasing the number of samples than can be extracted from_a given network and the expected network lifetime.

2.3.7 Effective Sample Rate

In a data collection network, effective sample rate is a primary application performance metric. We define the effective sample rate as the sample rate that sensor data can be taken at each individual sensor and communicated to a collection point in a data collection network. Fortunately, environmental data collection applications typically only demand sampling rates of 1-2 samples per minute. However, in addition to the sample rate of a single sensor, we must also consider the impact of the multi-hop networking architectures on a nodes ability to effectively relay the data of surrounding nodes. In a data collection tree, a node must handle the data of all of its descendents. If each child transmits a single sensor reading and a node has a total of 60 descendants, then it will be forced to transmit 60 times as much data. Additionally, it must be capable of receiving those 60 readings in a single sample period. This multiplicative increase in data communication has a significant effect on system requirements. Network bit rates combined with maximum network size end up impacting the effective per-node sample rate of the complete system .

One mechanism for increasing the effective sample rate beyond the raw communication capabilities of the network is to exploit in-network processing. Various forms of spatial and temporal compression can be used to reduce the communication bandwidth required while maintaining the same effective sampling rate. Additionally local storage can be used to collect and store data at a high sample rate for short periods of time. In-network data processing can be used to determine when an “interesting” event has occurred and automatically trigger data storage. The data can then be downloaded over the multi-hop network as bandwidth allows. Triggering is the simplest form of in-network processing. It is commonly used in security systems. Effectively, each individual sensor is sampled continuously, processed, and only when a security breach has occurred is data transmitted to the base station. If there were no local computation, a continuous stream of redundant sensor readings would have to be transmitted. We show how this same process can be extended to complex detection events.

2.4 Individual node evaluation metrics

2.4.1 Power

To meet the multi-year application requirements individual sensor nodes must be incredibly low-power. Unlike cell phones, with average power consumption measured in hundreds of milliamps and multi-day lifetimes, the average power consumption of wireless sensor network nodes must be measured in micro amps. This ultra-low-power operation can only be achieved by combining both low-power hardware components and low duty-cycle operation techniques.

During active operation, radio communication will constitute a significant fraction of the node's total energy budget. Algorithms and protocols must be developed to reduce radio activity whenever possible. This can be achieved by using localized computation to reduce the streams of data being generated by sensors and through application specific protocols. For example, events from multiple sensor nodes can be combined together by a local group of nodes before transmitting a single result across the sensor network. Our discussion on available energy sources will show that a node must consume less that 200 uA on average to last for one year on a pair of AA batteries. In contrast the average power consumption of a cell phone is typically more than 4000 uA, a 20 fold difference.

2.4.2 Flexibility

The wide range of usage scenarios being considered means that the node architecture must be flexible and adaptive. Each application scenario will demand a slightly different mix of lifetime, sample rate, response time and in-network processing. A wireless sensor network architecture must be flexible enough to accommodate a wide range of application behaviors. Additionally, for cost reasons each device will have only the hardware and software it actually needs for a given the application. The architecture must make it easy to assemble just the right set of software and hardware components. Thus, these devices require an unusual degree of hardware and software modularity while simultaneously maintaining efficiency.

2.4.3 Robustness

In order to support the lifetime requirements demanded, each node must be constructed to be as robust as possible. In a typical deployment, hundreds of nodes will have to work in harmony for years. To achieve this, the system must be constructed so that it can tolerate and adapt to individual node failure. Additionally, each node must be designed to be as robust as possible. System modularity is a powerful tool that can be used to develop a robust system. By dividing system functionality into isolated sub-pieces, each function can be fully tested in isolation prior to combining them into a complete application. To facilitate this, system components should be as independent as possible and have interfaces that are narrow, in order to prevent unexpected interactions. In addition to increasing the system's robustness to node failure, a wireless sensor network must also be robust to external interference. As these networks will often coexist with other wireless systems, they need the ability to adapt their behavior accordingly. The robustness of wireless links to external interference can be greatly increased through the use of multi-channel and spread spectrum radios. It is common for facilities to have existing wireless devices that operate on one or more frequencies. The ability to avoid congested frequencies is essential in order to guarantee a successful deployment.

2.4.4 Computation

The two most computationally intensive operations for a wireless sensor node are the in-network data processing and the management of the low-level wireless communication protocols. As we discuss later, there are strict real-time requirements associated with both communication and sensing. As data is arriving over the network, the CPU must simultaneously control the radio and record/decode the incoming data. Higher communication rates required faster computation. The same is true for processing being performed on sensor data. Analog sensors can generate thousands of samples per second. Common sensor processing operations include digital filtering, averaging, threshold detection, correlation and spectral analysis. It may even be necessary to perform a real-time FFT on incoming data in order to detect a high-level event. In addition to being able to locally process, refine and discard sensor readings, it can be beneficial to combine data with neighboring sensors before transmission across a network. Just as complex sensor waveforms can be reduced to key events, the results from multiple nodes can be synthesized together. This in-network processing requires additional computational resources. In our experience, 2-4 MIPS of processing are required to implement the radio communication protocols used in wireless sensor networks. Beyond that, the application data processing can consume an arbitrary amount of computation depending on the calculations being performed.

2.4.5 Time Synchronization

In order to support time correlated sensor readings and low-duty cycle operation of our data collection application scenario, nodes must be able to maintain precise time synchronization with other members of the network. Nodes need to sleep and awake together so that they can periodically communicate. Errors in the timing mechanism will create inefficiencies that result in increased duty cycles.

In distributed systems, clocks drift apart over time due to inaccuracies in timekeeping mechanisms. Depending on temperature, voltage, humidity, time keeping oscillators operate at slightly different frequencies. High-precision synchronization mechanisms must be provided to continually compensate for these inaccuracies.

2.4.6 Size & Cost

The physical size and cost of each individual sensor node has a significant and direct impact on the ease and cost of deployment. Total cost of ownership and initial deployment cost are two key factors that will drive the adoption of wireless sensor network technologies. In data collection networks, researchers will often be operating off of a fixed budget. Their primary goal will be to collect data from as many locations as possible without exceeding their fixed budget. A reduction in per-node cost will result in the ability to purchase more nodes, deploy a collection network with higher density, and collect more data. Physical size also impacts the ease of network deployment. Smaller nodes can be placed in more locations and used in more scenarios. In the node tracking scenario, smaller, lower cost nodes will result in the ability to track more objects.

2.5 Hardware Capabilities

Now that we have identified the key characteristics of a wireless sensor node we can look at the capabilities of modern hardware. This allows us to understand what bit rate, power consumption, memory and cost we can expect to achieve. A balance must be maintained between capability, power consumption and size in order to best address application needs. This section gives a quick overview of modern technology and the tradeoffs between different technologies. We start with a background of energy storage technologies and continue through the radio, CPU, and sensors.

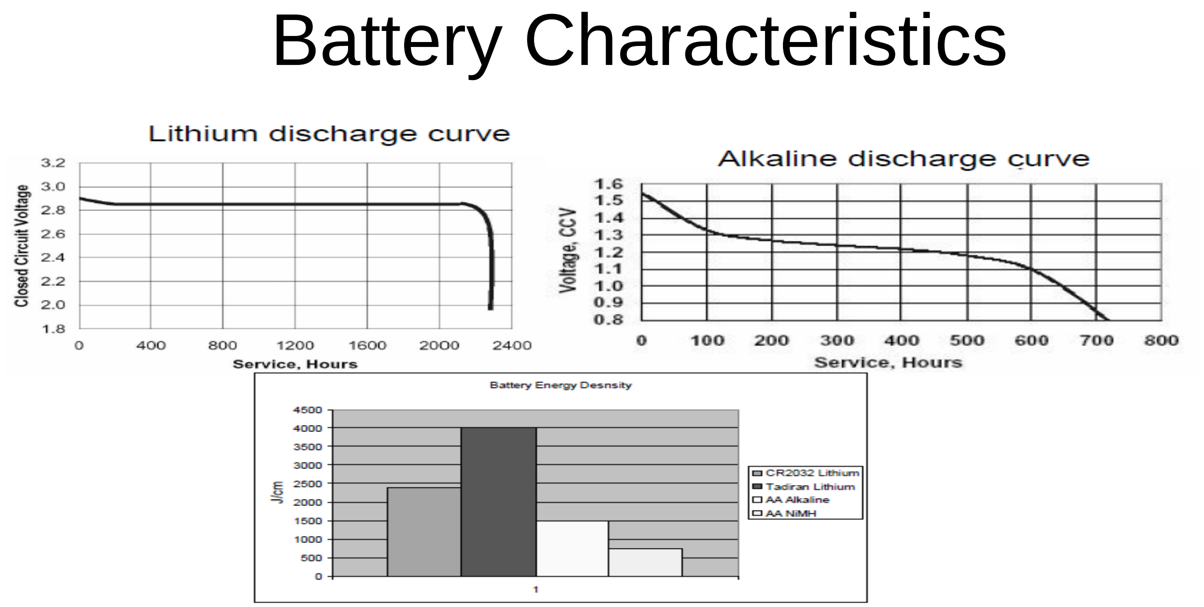
2.5.1 Energy

Just as power consumption of system components are often expressed in milliamps, batteries are generally rated in milliamp-hours (mAh). In theory a 1000 mAh battery could support a processor consuming 10 mA for 100 hours. In practice this in not always true. Due to battery chemistry, voltage and current levels vary depending on how the energy is extracted from a battery. Additionally, as batteries discharge their voltage drops. If the system is not tolerant to a decrease in voltage it may not be possible to use the full rated capacity of a battery. For example, a 1.5 V alkaline battery is not considered empty by the manufacturer until it is outputting only 8 V

2.5.1.1 Battery technologies

There are three common battery technologies that are applicable for wireless sensor networks – Alkaline, Lithium, and Nickel Metal Hydride. An AA Alkaline battery is rated at 1.5 V, but during operation it ranges from 1.65 to .8 V as shown in Figure 8 and is rated at 2850 mAh. With a volume of just 8.5 cm3, it has an energy density of approx 1500 Joules/cm3. While providing a cheap, high capacity, energy source, the major drawbacks of alkaline batteries are the wide voltage range that must be tolerated and their large physical size. Additionally, lifetimes beyond 5 years cannot be achieved because of battery self-discharge. The shelf-life of an alkaline battery is approximately 5 years.

Lithium batteries provide an incredibly compact power source. The smallest versions are just a few millimeters across. Additionally, they provide a constant voltage supply that decays little as the battery is drained. Devices that operate off of lithium batteries do not have to be as tolerant to voltage changes as devices that operate off of alkaline batteries. Additionally, unlike alkaline batteries, lithium batteries are able to operate at temperatures down to -40 C. The most common lithium battery is the CR2032 . It is rated at 3V, 255 mAh and sells for just 16 cents. With a volume of 1 cm3, it has and energy density of 2400 J/cm3. In addition to traditional lithium batteries, there are also specialized Tadiran lithium batteries that have densities as high as 4000 J/cm3 and tolerate a wide temperature range. One of the drawbacks of lithium batteries is that they often have very low nominal discharge currents.



Battery Characteristics

Figure 8: Battery characteristics for Lithium, Alkaline and NiMH batteries. The discharge_characteristics of alkaline batteries make it essential to design a system to tolerate a wide range of input voltages.

supply that decays little as the battery is drained. Devices that operate off of lithium batteries do not have to be as tolerant to voltage changes as devices that operate off of alkaline batteries. Additionally, unlike alkaline batteries, lithium batteries are able to operate at temperatures down to -40 C. The most common lithium battery is the CR2032
. It is rated at 3V, 255 mAh and sells for just 16 cents. With a volume of 1 cm3, it has and energy density of 2400 J/cm3. In addition to traditional lithium batteries, there are also specialized Tadiran lithium batteries that have densities as high as 4000 J/cm3
and tolerate a wide temperature range. One of the drawbacks of lithium batteries is that they often have very low nominal discharge currents. A D-cell size Tadiran battery has a. nominal discharge current of just 3 mA. This is compared to an alkaline AA battery's nominal discharge rate of 25 mA. Nickel Metal Hydride batteries are the third major battery type. They have the benefit of being easily rechargeable. The downside to rechargeable batteries is a significant decrease in energy

density. An AA size NiMH battery has approximately half the energy density of an alkaline battery at approximately 5 times the cost. Before considering the use of NiMH batteries it is important to note that they only produce 1.2V. Because many system components require 2.7 volts or more, they it may not be possible to operate directly off of rechargeable batteries.

2.5.1.2 Expected lifetime calculation

While it appears easy to quickly look at a battery's rated capacity, compare it to the systems energy consumption and calculate the system lifetime, there are several additional factors to consider.

In looking at real systems, it is important to look at how power supplies decay over time. Figure 2-1 shows the voltage versus time plot of an AA battery drained at a 500 mW. The graph shows that the battery quickly falls from the 1.5 V starting voltage and ends at just .8 V. If a theoretical system was built with components requiring 2.7 V and consumed 250 mW, it would only last for 100 minutes off of 2 AA batteries. However, if the system components were selected to operate off of voltages down to 2.0 volts, it would last approximately 5 times as long off of the same power source. A seemingly unimportant CPU parameter results in a 5x difference in system lifetime.

2.5.1.3Renewable Energy

An alternative to relying on batteries with enough energy to last for years is to use renewable energy. Modern solar cells can produce up to 10 mW per square inch in direct sunlight. If stored properly, the energy collected during the day can be enough energy to last through the night. In indoor lighting environments between 10 and 100 uW per square inch can be produced depending on the type of lighting. For solar powered application scenarios, the key to successfully harnessing solar energy lies in the ability to store the energy. Modern ultra-capacitors represent an option for energy storage. Ultra capacitors are low-voltage capacitors ranging between 1 and 6 Farads. When charged to 3 V they contain enough energy to send several hundred radio packets. While they can be charged and drained easily, one of their biggest drawbacks is that they have internal leakage rates of 20 to 50 uA.

Chapter -3

Energy efficiency

The diversity of the application supported by wireless ad hoc and sensor networks explain the success of these networks. However, nodes in such networks can have a limited amount of energy. This energy can be very expensive, difficult or even impossible to renew. So saving energy to maximize network lifetime is one of the critical problems in wireless ad hoc and sensor networks. That is why algorithms and protocols operating in such networks should be energy efficient. Several solutions are proposed to improve the network lifetime

3.1 Definition of network lifetime

All energy efficient techniques share the same goal: to maximize network lifetime. Unfortunately, there is no definition of network lifetime commonly agreed in the literature. Several definitions of network lifetime exist, the most frequently used are:

_ Definition D1: Time to first node failure due to battery outage. As sensor redundancy is generally used, a sensor failure can have no influence on the network and application functionalities. That is why, some authors prefer the time to the failure of a certain percentage of the sensors(e.g. 20%), in order to take into account possible redundancy.

_ Definition D2: Time to application failure: an application functionality is no longer ensured_Definition D1 differs from definitionD2 because of redundancy in sensor coverage. Indeed, if an area is covered by k sensors, the failure of k-1 of them is perfectly tolerated.

_ Definition D3: Time to first network partitioning. As soon as the network is no longer connected, vital information can no longer be transferred to its destination. In the absence of knowledge of the application supported by the network, definitions D1 and D3 are the most useful ones to compare different energy efficient strategies. In all cases, an energy consumption model is needed to conclude in favor of an increase in network lifetime. In next section, we present the energy consumption model the most frequently used.

3.2 Energy consumption model

There are many energy consumption models proposed in the literature. We can unified [these models by the following one highlighting two components of the energy dissipated by the transmitter. The first component reflects the energy consumed by the radio. The second component presents the energy consumed by the amplifier and depends on the distance between the transmitter and the receiver.

E transmit = C1(size) + C2(size; d) = C1 _ size + C2 _ size _ d_ = size(C1 + C2 _ d_); (3.2) with: C1: Energy consumed by the radio of the transmitter to transmit a bit, C2: Energy consumed by the amplifier to send a bit at a distance of 1 meter, size: Packet size,
d: Distance between the transmitter and the receiver,
and 0 < _ < 6 values of 2 or 4 are the most frequently used.
Many works about topology control focus on the component proportional to the distance. Equation 3.2 becomes, when uniformed by the size of the transmitted packet:
$E_{transmit}$ = C1 + C2 _ d_.
This formula points out the relation between energy consumption and distance. This relation is used in topology control to optimize energy consumption by tuning the transmission power taking into account the distance between the transmitter and the receiver. Many other works suppose that the transmissions is done at the maximum power. In other words, the transmitter uses the transmission power such that any receiver at a distance equal to the transmission rage correctly receives

the message. Consequently, we can consider the quantity (C1 + C2 _ d_) as a constant named C. Hence, the energy dissipated in a transmission by a transmitter is : $E_{transmit}$ = C _ size where size denotes the packet size in bits.

In our work, we assume that: the transmission power is a constant and the same for all nodes in the network. Concerning the receiver, it consumes energy to capture packets. this energy is expressed as follows: $E_{receive}$ = C1(size) = C1 _ size; 3.2 with: C1: Energy consumed by the radio of the receiver to capture one bit, size: Packet size. We notice that this energy can not be neglected in the energy dissipated during transmission

3.3 Energy consumption in wireless networks

In addition to the transmit and receive states a wireless node can be in the idle or sleep states. In next section we present and explain each wireless node state.

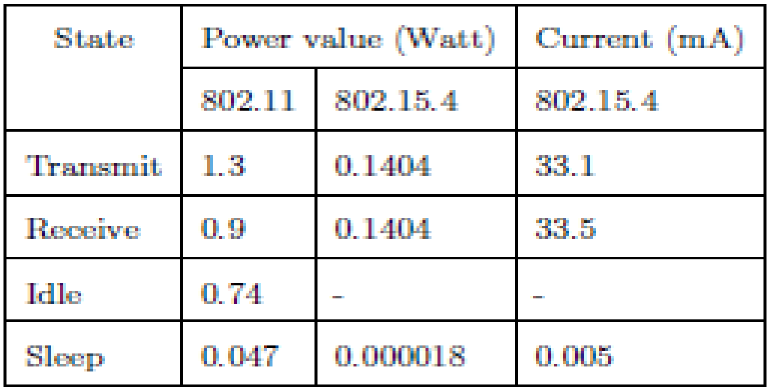
3.3.1 Energy states

A wireless node can be in one of the following four states: Transmit, Receive, Idle or Sleep. Each state corresponds to a different power level (see Table 1):

_ **Transmit:** node is transmitting a frame with transmission power $P_{transmit}$;
_ Receive: node is receiving a frame with reception power $P_{receive}$. This frame can be decoded
by this node or not, it can be intended to this node or not;

_ **Idle (listening):** even when no messages are being transmitted over the medium, the nodes
stay idle and keep listening the medium with P_{idle};

_ **Sleep:** the radio is turned off, and the node is not capable of detecting signals: no communication is possible. The node uses Psleep that is largely smaller than in any other state: the energy consumption is minimum;



State	Power value (Watt)		Current (mA)
	802.11	802.15.4	802.15.4
Transmit	1.3	0.1404	33.1
Receive	0.9	0.1404	33.5
Idle	0.74	-	-
Sleep	0.047	0.000018	0.005

Table 1: Power value in each radio state.

In Table 1, we report the reference values of power consumption in each state taken from a Lucent silver wave lan PC card [58] implementing the IEEE 802.11b medium access and a ZigBee node implementing IEEE 802.15.4medium access. In both cases, we can notice that the least consuming state is the sleep state. However, the power used in transmit and receive state is close for the IEEE 802.15.4 medium access.

3.3.2 Reasons of energy consumption in the network

In wireless ad hoc and sensor networks, nodes dissipate energy in processing, transmitting and receiving messages. This energy is needed for correct working of the wireless networks. In addition to this energy, there is a great amount of energy wasted in states that are useless from the application point of view, such as:

_ idle listening: since a node does not know when it will receive a message it must permanently
listen to the medium and so it remains in the idle state. As we can notice in Table 2.1 the power used in idle state is close to the power used in receive state.
_ overhearing: When a sender transmits one packet to next hop, because of the shared nature of wireless medium, all neighbors of the source receive this packet even if it is intended to only one of them. Thus the overhearing is the energy dissipated when the node is an one-hop neighbor of the sender and is not the destination.
_ interference: Each node situated between transmitter range and interference range receives this packet but it cannot decode it.
_ collision: In case of CSMA/CA medium access, When a collision occurs, the energy dissipated for the transmission and for the reception of colliding frames is wasted. The energy constrained nature of wireless nodes requires the use of energy efficient strategies to minimize the energy wasted in these useless states and so maximize network lifetime. In the next section, we describe and classify works aimed at minimizing energy consumption and improving network lifetime.

3.4 Reduce the amount of information transferred

The last energy efficient strategy consists in aggregating information (e.g.; in a data gathering application, a node sends to its parent a single message containing the values transmitted by its children), reducing wasteful transmissions (e.g.; transmission of an information that is already known by the receiver or in which it has no interest) or tuning the refreshment period of control messages (e.g.; neighborhood discovery, topology dissemination, data gathering tree structure). These solutions can be classified in three classes:
_ Information aggregation,
_ Optimized flooding,
_ Tuning the information refreshment frequency.
In the following, we present each class by quoting some works focusing in each class.

Chapter- 4

BPSK , QPSK, 16-QAM, 64-QAM modulation and Fading channels

4.1.1 BPSK

Binary phase-shift keying (**BPSK**) Constellation diagram example for BPSK. BPSK (also sometimes called PRK, phase reversal keying, or 2PSK) is the simplest form of phase shift keying (PSK). It uses two phases which are separated by 180° and so can also be termed 2-PSK.

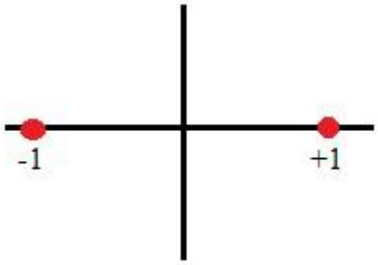


Fig 8: BPSK Costellation Diagram

4.1.2 QPSK

Quadrature Phase Shift Keying is type of phase shift keying. Unlike BPSK which is a DSBCS modulation scheme with digital information for the message, **QPSK** is also a DSBCS modulation scheme but it sends two bits of digital information a time (without the use of another carrier frequency).

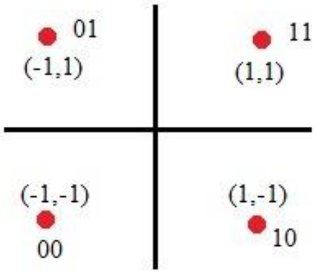


Fig 9: QPSK Costellation Diagram

4.2 QAM

Quadrature amplitude modulation is widely used in many digital data radio communications and data communications applications. A variety of forms of QAM are available and some of the more common forms include 16 QAM, 32 QAM, 64 QAM, 128 QAM, and 256 QAM. Here the figures refer to the number of points on the constellation, i.e. the number of distinct states that can exist.

The various flavors of QAM may be used when data-rates beyond those offered by 8-PSK are required by a radio communications system. This is because QAM achieves a greater distance between adjacent points in the I-Q plane by distributing the points more evenly. And in this way the points on the constellation are more distinct and data errors are reduced. While it is possible to transmit more bits per symbol, if the energy of the constellation is to remain the same, the points on the constellation must be closer together and the transmission becomes more susceptible to noise. This results in a higher bit error rate than for the lower order QAM variants. In this way there is a balance between obtaining the higher data rates and maintaining an acceptable bit error rate for any radio communications system.

4.2.1 QAM applications

QAM is in many radio communications and data delivery applications. However some specific variants of QAM are used in some specific applications and standards.

For domestic broadcast applications for example, 64 QAM and 256 QAM are often used in digital cable television and cable modem applications. In the UK, 16 QAM and 64 QAM are currently used for digital terrestrial television using DVB - Digital Video Broadcasting. In the US, 64 QAM and 256 QAM are the mandated modulation schemes for digital cable as standardised by the SCTE in the standard ANSI/SCTE 07 2000.

In addition to this, variants of QAM are also used for many wireless and cellular technology applications.

4.2.2Constellation diagrams for QAM

The constellation diagrams show the different positions for the states within different forms of QAM, quadrature amplitude modulation. As the order of the modulation increases, so does the number of points on the QAM constellation diagram.

The diagrams below show constellation diagrams for a variety of formats of modulation:

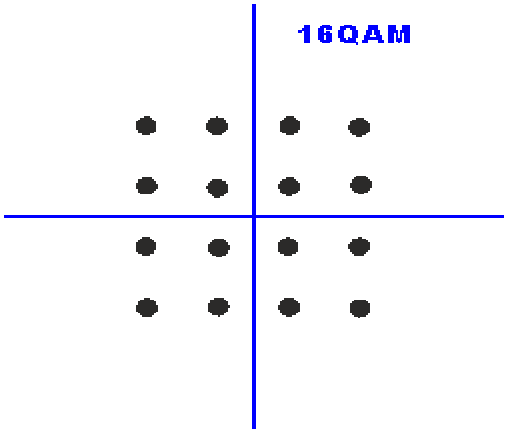


Fig 10: 16-QAM Constellation Diagram

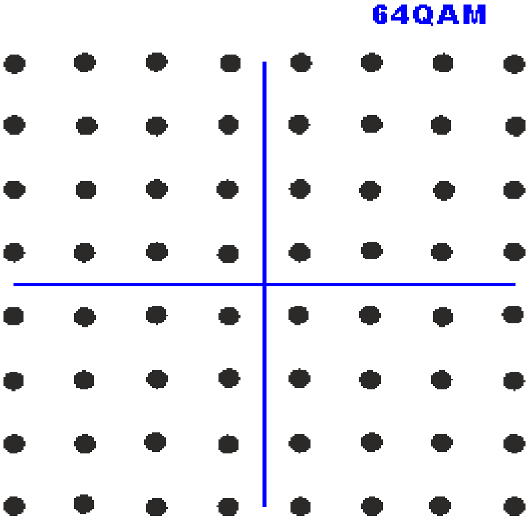


Fig 11: 16-QAM Constellation Diagram

4.2.3 QAM bits per symbol

The advantage of using QAM is that it is a higher order form of modulation and as a result it is able to carry more bits of information per symbol. By selecting a higher order format of QAM, the data rate of a link can be increased.

The table below gives a summary of the bit rates of different forms of QAM and PSK.

Table 2:

Modulation	Bits per symbol	Symbol Rate
BPSK	1	1 x bit rate
QPSK	2	1/2 bit rate
16QAM	4	1/4 bit rate
64QAM	6	1/6 bit rate

4.2.4 QAM noise margin

While higher order modulation rates are able to offer much faster data rates and higher levels of spectral efficiency for the radio communications system, this comes at a price. The higher order modulation schemes are considerably less resilient to noise and interference.

As a result of this, many radio communications systems now use dynamic adaptive modulation techniques. They sense the channel conditions and adapt the modulation scheme to obtain the highest data rate for the given conditions. As signal to noise ratios decrease errors will increase along with re-sends of the data, thereby slowing throughput. By reverting to a lower order modulation scheme the link can be made more reliable with fewer data errors and re-sends.

4.3.1 PSK transceiver model

The architecture of PSK transceiver is more complex than OOK transceiver, which is shown in Figure 13. The direct-conversion transmitter is also adopted in the PSK system, in which the phase modulation is accomplished by the phase-shifting-network (PSN) controlled by digital baseband signal. The modulation signal is transmitted directly to the PA without crossing the mixer. The heterodyne low-IF receiver is adopted in PSK systems. The main components in the receiver signal chain are the RF filter, LNA, downconversion mixer, baseband amplifier, baseband, and antialiasing filter, and ADC

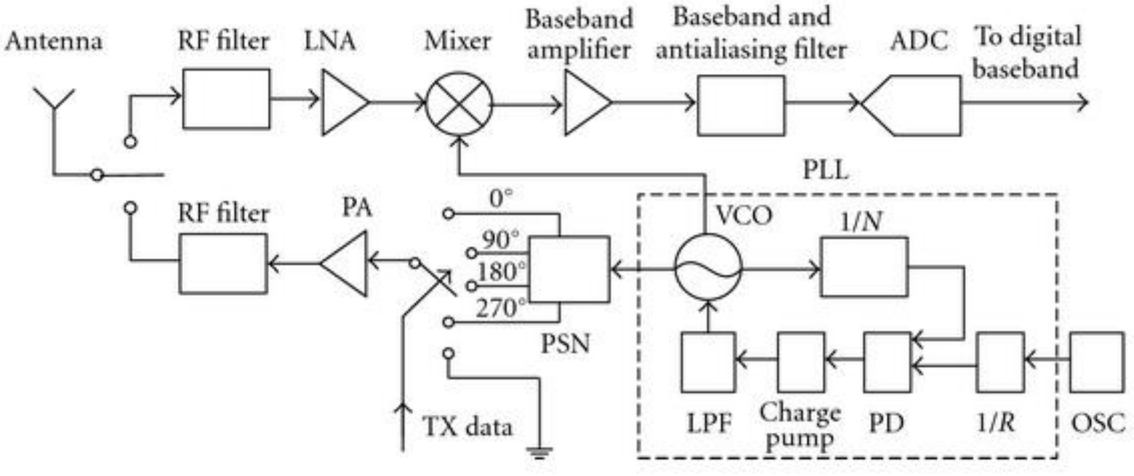


Figure 13: psk transceiver model

4.3.2 QAM transceiver model

Figure 14 describes the RF front end of QAM system, which also employs the same receiver and direct-conversion transmitter. The main components of the analog signal chain of the transmitter are digital-to-analog converter (DAC), reconstruction filter, up conversion mixer, power amplifier (PA), and RF filter.

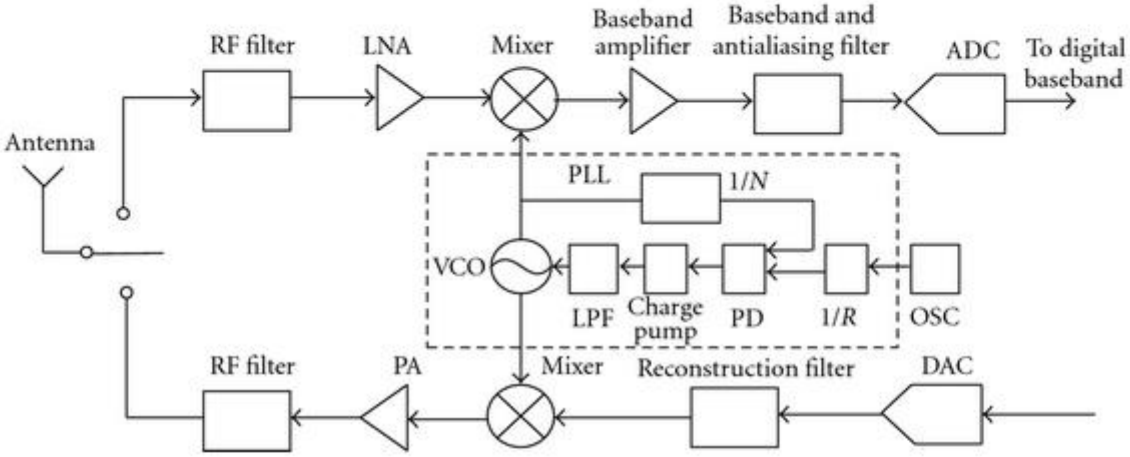


Figure 14: QAM transceiver model

4.4.1 AWGN noise

nodes and surrounding objects were static, with minimal time-varying changes in the wireless channel. In this case, the wireless channel is well described by an additive white Gaussian noise (AWGN) model. AWGN is a noise that affects the transmitted signal when it passes through the channel. It contains a uniform continuous frequency spectrum over a particular frequency band.

A basic and generally accepted model for thermal noise in communication channels, is the set of assumptions that

- the noise is additive, i.e., the received signal equals the transmit signal plus some noise, where the noise is statistically independent of the signal.
- the noise is white, i.e, the power spectral density is flat, so the autocorrelation of the noise in time domain is zero for any non-zero time offset.
- the noise samples have a Gaussian distribution.

Mostly it is also assumed that the channel is Linear and Time Invariant. The most basic results further assume that it is also frequency non-selective.

In this paper, we determine how to minimize energy consumption per information bit in a single link, with the consideration of packet retransmission and overhead. This is achieved by deriving expressions for the optimum target bit error probability and packet length at different transmission distances. Furthermore, the energy consumptions of different modulation schemes are compared over an additive white Gaussian noise (AWGN) channel. Finally, it is shown that the optimum target bit error probability and packet length converge to a constant value for long distances. Numerical results show that at short distances, it is optimum to use bandwidth efficient modulation with large packets and low target BER, and at long distances, it is optimum to use energy efficient modulation with short packets and high target BER.

4.4.2 Rayleigh Fading

When no LOS path exists in between transmitter and receiver, but only have indirect path than the resultant signal received at the receiver will be the sum of all the reflected and scattered waves. In Rayleigh fading obstacles were moved within the network, along a line of 20m. Furthermore, a metal object was put in front of the edge node, so the edge node and the relays were not in line-of-sight. The edge node was moved on a distance of few tens of centimeters.

This work is motivated by the problem of characterizing small-scale radio propagation environments for wireless sensor networks. If sensors are statically deployed near the ground or within structures, temporal fading may not exist but the channel may nevertheless experience severe frequency-selective behavior. The work presents real-world, frequency-selective fading data measured for in-vehicle wireless sensor applications.

Chapter- 5

Optimal SNR and Life time Transceiver model

When the communication system is power- limited (as in WSN), the common notion is to choose low- order modulations such as BFSK or BPSK, which has a low SNR requirement for achieving a desired bit error rate [4]. These modulations are, in fact, the ones used in commercially available low-power transceivers like the TI CC1000 [5] or CC2420 [6].often used for WSN applications. Nevertheless it has been shown that the above notion leads to suboptimal operation for short link distances.

We have tried to determine the total energy that is necessary for transferring one bit of data successfully, without error, in a point-to-point packet-switched wireless communication link (e.g. between two sensor nodes). We assume that every frame transmitted in the forward direction is matched by a feedback frame in the *reverse* direction, which acknowledges correct reception or requests a re-transmission(10). We also assume that the irradiated power is determined by the transmitter based upon knowledge of the statistics of the signal-to-noise ratio (SNR) at the decision stage of the intended receiver. We further assume that all frames in both directions are always detected and that all feedback frames are decoded without error. That's why we have determined the Optimal SNR values for different random channel model.

A. Optimal SNR

For obtaining Optimal SNR we have used the Minimum Optimal SNR Equation[11].

$$\lambda\left(\frac{P_{\mathrm{el}}}{A_{\mathrm{total}}}+\bar{\gamma}_{\mathrm{o}}\right)\frac{d\bar{P}_{\mathrm{s}}}{d\bar{\gamma}}(\bar{\gamma}_{\mathrm{o}})-\bar{P}_{\mathrm{s}}(\bar{\gamma}_{\mathrm{o}})+1=0 \qquad (101)$$

It is to be noted that the only parameters that influence Minimum Optimal SNR level $\bar{\gamma}_0$are the mean symbol error rate, $\bar{P}(\bar{\gamma})$,the number of payload symbols per frame, λ ,and the ratio between the power consumption of electronic components, P_{el}, and the coefficient A_{total}, which is proportional to the irradiated power. Equation (101) can be used to find the optimal SNR for different random channel models(12).

B. Transceiver model

To illustrate transceiver life time, let us consider a simple network composed by two wireless sensor node switch parameters as given in Table I. The nodes exchange 20 kbits of data every 3 minutes. Each node is powered by an ideal 1.2 Volt AA battery with a 2000 mAh initial energy charge.

Table 3. Low Power Device Parameters

Parameter	Description	Value
R_S	Symbol rate	20 kBaud
L	Frame Payload	98bits
O	Overhead	30bits
E_{st}	Start-up energy	0.125 nJ
A	Path-loss coefficient	3.5
A	Channel loss	30 dB
H	PA efficiency	0.35%
$P_{el,tx}$	Tx electric power consumption	98.2 mW
$P_{el,rx}$	Rx electric power consumption	112.5 mW
M_l	Link margin	40 dB

Symbol rate:

In digital communications, **symbol rate**, also known as **modulation rate**, is the number of symbol changes, waveform changes, or signaling events, across the transmission medium per time unit using a digitally modulated signal or a line code. The symbol rate is measured in baud (Bd) or symbols per second. In the case of a line code, the symbol rate is the pulse rate in pulses per second. Each symbol can represent or convey one or several bits of data. The symbol rate is related to the gross bitrate expressed in bits per second.

Frame pay load:

A data packet on an Ethernet link is called an Ethernet packet, which transports an Ethernet frame as its payload. An Ethernet frame is preceded by a preamble and start frame delimiter (SFD), which are both part of the Ethernet packet at the physical layer.

Traffic overhead:

In wireless sensor networks (WSN) data produced by one or more sources usually has to be routed through several intermediate nodes to reach the destination. Problems arise when intermediate nodes fail to forward the incoming messages. This is called Traffic Overhead.

Path loss:

Path loss (or **path attenuation**) is the reduction in power density (attenuation) of an electromagnetic wave as it propagates through space. Path loss is a major component in the analysis and design of the link budget of a telecommunication system.

This term is commonly used in wireless communications and signal propagation. Path loss may be due to many effects, such as free-space loss, refraction, diffraction, reflection,aperture-medium coupling loss, and absorption. Path loss is also influenced by terrain contours, environment (urban or rural, vegetation and foliage), propagation medium (dry or moist air), the distance between the transmitter and the receiver, and the height and location of antennas.

the path loss co-efficient is the calculated value of "free air" obstructions such as dust, humidity etc vs the frequency band being calculated for. Specifically things like rain fade would be one of the factors used in this parameter.

Channel loss:

This Operation accounts for losses or gains of a operation that occur along a channel reach as a result of flow through the channel bottom and evaporation from the stream surface. Even though channel losses are actually distributed along the length of the reach the Operation adjusts instantaneous discharges at a flow point for such losses.

Chapter - 6

Simulation Results & Analysis:

A. Simulation

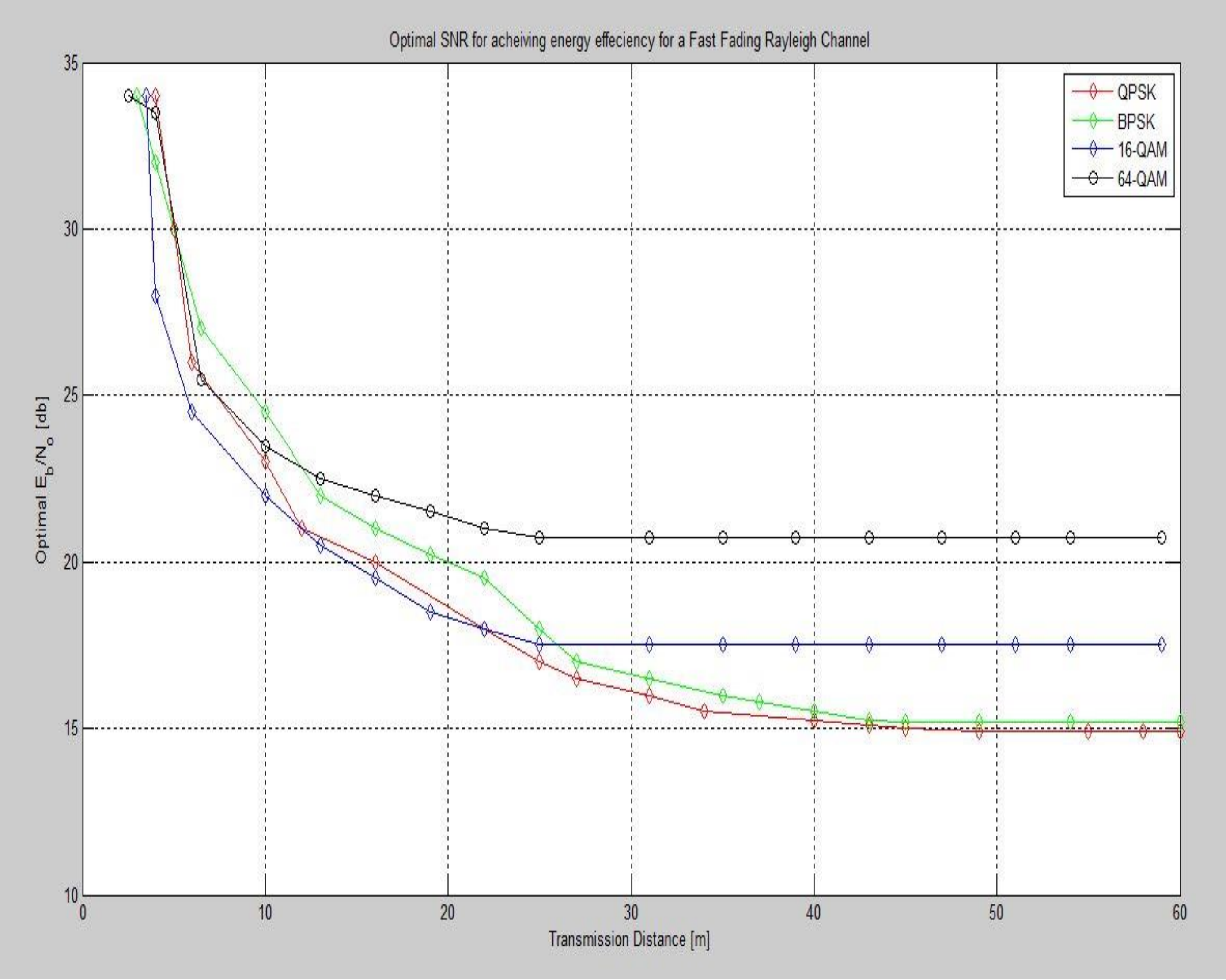


Fig. 15. Optimal SNR for achieving energy efficiency as function of link distance for a fast fading Rayleigh channel.

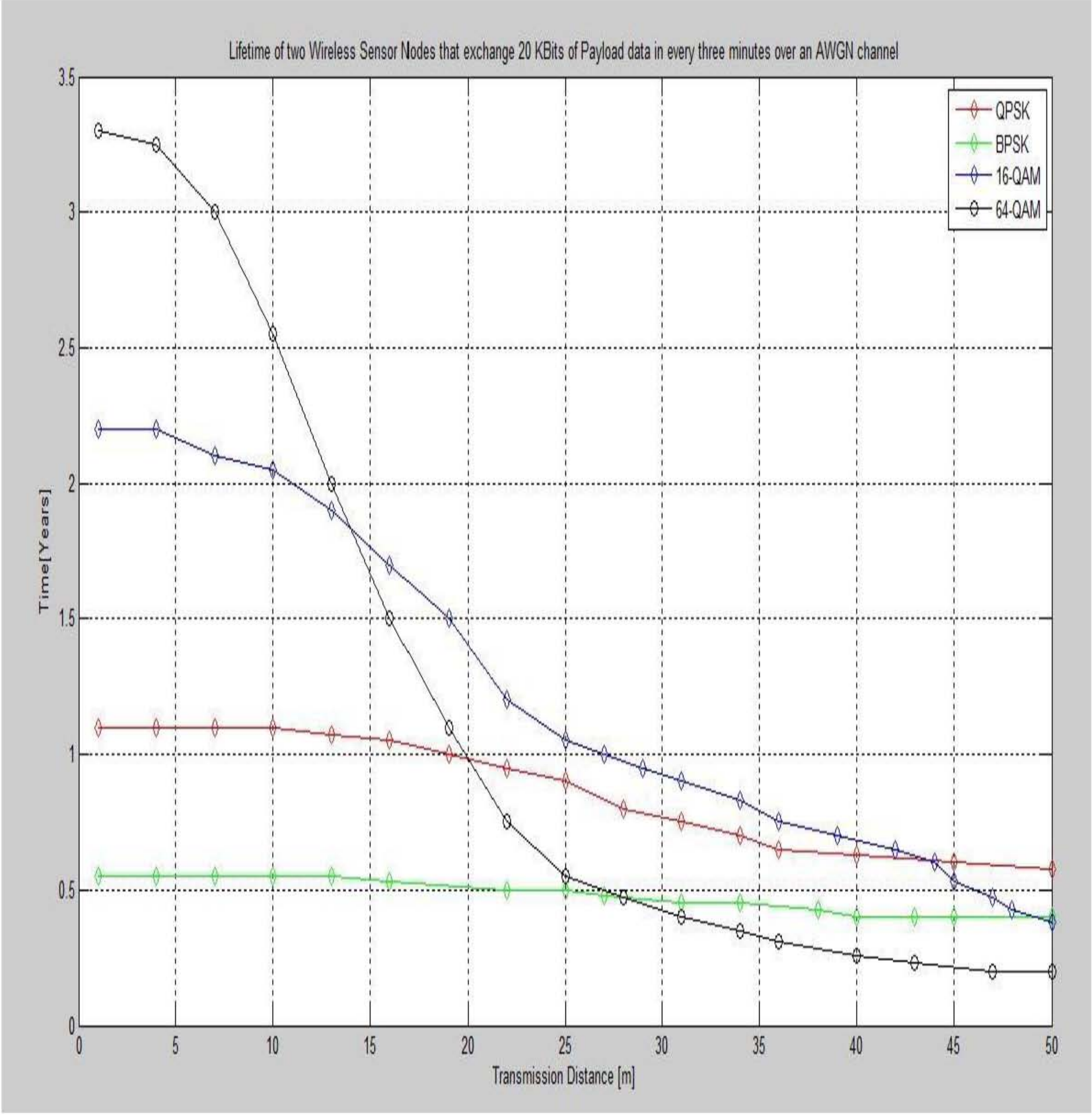


Fig.16 : Lifetime of two wireless sensor nodes that exchange 20 Kbits of Pay load

data every three minutes over an AWGN Channel

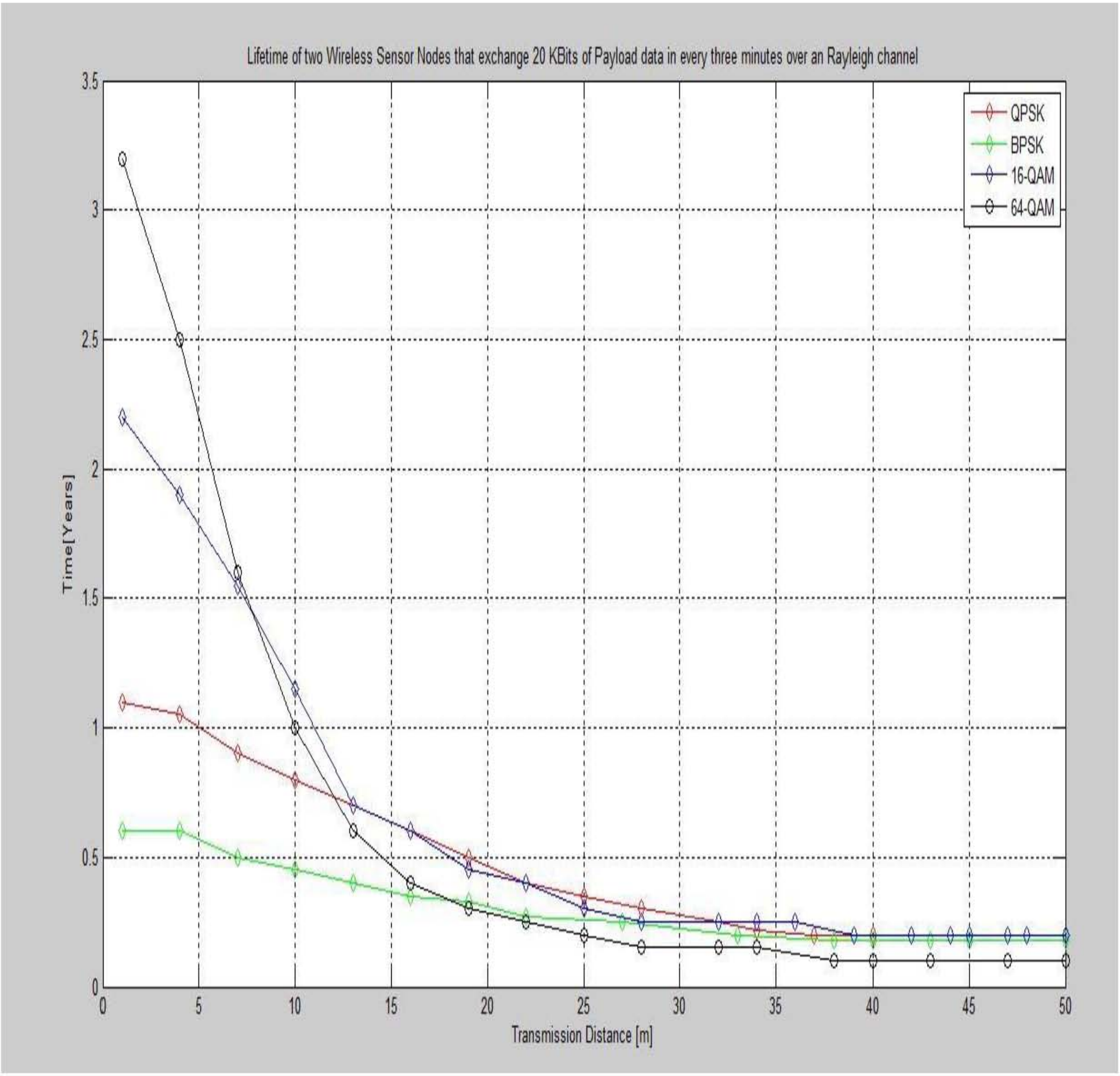


Fig.17: : Lifetime of two wireless sensor nodes that exchange 20 Kbit of Pay load data every three minutes over an Rayleigh Channel

B. Analysis

Numerical evaluations of (101) using the parameters presented in Table I show that BPSK, QPSK and various M- QAM modulations attains it minimum energy consumption at a different SNR. As can be seen in table 3, the SNR at which these minima occur varies with transmission distance (curves are plotted against E_b/N_0 to compare the results against an equal amount of energy per bit).

From Figure 15 & 16 ,using the above mentioned model, the average lifetime of the batteries of these two nodes was calculated for BPSK, QPSK and M-QAM transmissions over different channel models as a function of link distance, with each modulation operated at its optimal SNR. It was found that as distance decreases, the longest network lifetime is achieved by more spectrally efficient modulations (Figure 14 for the AWGN channel and Figure 16 for fast fading Rayleigh channel). It is apparent that, regardless of the channel type, lifetime extensions up to 550% can be gained in short range networks by selecting modulations with larger constellations than BPSK.

Chapter - 7

Conclusion:

We have detected that for a given modulation scheme the average energy consumed per bit by transmissions over a fast fading channel as function of the SNR has a unique minimum value, which is obtained at an SNR which is optimal in the energy consumption sense. The parameters that influence this optimal SNR are the mean symbol error rate, the number of payload symbols per transmission frame and the ratio between the power consumption of electronic components versus the irradiated power.

We also found that for long transmission distances, low bandwidth efficiency modulations (small M -ary number, like BPSK) are optimal in the energy consumption sense. As the transmission distance shortens the optimal modulation size grows. In short range communications the power consumed by electronic components dominates over the irradiated power, and hence also does so over the energy consumption of the power amplifier. Under these conditions the average air time spent per data bit becomes a relevant parameter in the total energy budget. This makes optimal to pack more bits into each symbol and thereby to choose a larger modulation size.

Lastly ,our results show that lifetime extensions up to 550% can be gained in short range networks by selecting modulations with larger constellations than BPSK.

Recommendation for Future Research

Those who are keen to work in this topic, can work over attack on wireless sensor network by understanding its characteristics and basic model.

References:

- [1] F.Akyildiz I F, Su W and Sankarsubramaniam Y 2002 Wireless sensor networks: a survey Computer Networks 38393-422
- [2] J. N. Al-Karaki and A. E. Kamal, “Routing Techniques in WSNs. IEEE Communications Magazine”, **(2004)**December.
- [3] R. Verdone, D. Dardari, G. Mazzini, and A. Conti, *Wireless Sensor and Actuator Network: technologies, analysis and design*. Academic Press, 2008.
- [4] B. Sklar, *Digital Communications: Fundamentals and Applications*.
- Upper Saddle River, NJ, USA: Prentice Hall, 2001.
- [5] Chipcon, “CC1000 Single Chip Very Low Power RF Transceiver,” datasheet,2002.
- [6]“2.4 Ghz IEEE 802.15.4 Zigbee-ready RF Transceiver,” SWRS041 datasheet,2006.
- [7] V. Raghunathan, C. Schurgers, S. Park, and M. B. Srivastava, “Energy-aware wireless microsensor networks,” *IEEE Signal Processing Maga- zine*, vol. 19, no. 2, pp. 40–50, Mar. 2002.
- [8] S. Cui, A. J. Goldsmith, and A. Bahai, “Energy-constrained modulation optimization,” *Wireless Communications, IEEE Transactions on*, vol. 4, no. 5, pp. 2349–2360, 2005.
- [9] T. Wang, W. Heinzelman, and A. Seyedi, “Minimization of transceiver energy consumption in wireless sensor networks with AWGNchannels,” in *Communication, Control, and Computing, 46th Annual Allerton Conference on*, 2008, pp.62–66.
- [10]M. Holland, “Optimizing physical layer parameters for wireless sensor networks,” Master’s thesis, Rochester University,2007
- [11] Fernando Rosas “Modulation Optimization for achieving energy efficient Communication”*IEEE Signal Processing Magazine*, vol. 12, 2012.
- [12] K. Hartwell, G. G. Messier, and R. J. Davies, “Optimizing Phys- ical Layer Energy Consumption for Wireless Sensor Networks,” in *VTC2007-Spring*, 2007, pp.76–79.
- [13] Saoucene Mahfoudh “Energy efficiency in wireless ad hoc and sensor networks:routing, node activity scheduling and cross-layering
- [14] Jason Lester Hill “System Architecture for Wireless Sensor Networks”

Appendix A:

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% SEP: A Stable Election Protocol for clustered                        %
%      heterogeneous wireless sensor networks                          %
% (c) Georgios Smaragdakis                                             %
% WING group, Computer Science Department, Boston University           %
% You can find full documentation and related information at:          %
% http://csr.bu.edu/sep                                                %
% To report your comment or any bug please send e-mail to:             %
% gsmaragd@cs.bu.edu                                                   %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% This is the LEACH [1] code we have used.                             %
% The same code can be used for FAIR if m=1                            %
% [1] W.R.Heinzelman, A.P.Chandrakasan and H.Balakrishnan,             %
%     "An application-specific protocol architecture for wireless      %
%      microsensor networks"                                           %
%     IEEE Transactions on Wireless Communications, 1(4):660-670,2002  %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

clear;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% PARAMETERS %%%%%%%%%%%%%%%%%%%%%%%%%%%%

%Field Dimensions - x and y maximum (in meters)
xm = 100;
ym = 100;
%x and y Coordinates of the Sink
%sink.x =0.5 * xm;
%sink.y = ym + 50;
sink.x=50;
sink.y=175;
%sink.x=0.5*xm;
%sink.y=0.5*ym;

%Number of Nodes in the field
n = 100
%Optimal Election Probability of a node to become cluster head
p=0.05;
packetLength =6400;%Êý¾Ý°ü³¤¶È
ctrPacketLength = 200;%¿ØÖÆ°ü³¤¶È
%Energy Model (all values in Joules)
%Initial Energy
Eo = 0.5;
%Eelec=Etx=Erx
ETX=50*0.000000001;
ERX=50*0.000000001;
%Transmit Amplifier types
```

```
Efs=10*0.000000000001;
Emp=0.0013*0.000000000001;
%Data Aggregation Energy
EDA=5*0.000000001;

INFINITY = 999999999999999;
%maximum number of rounds
rmax=9999
%%%%%%%%%%%%%%%%%%%%%%%%% END OF PARAMETERS %%%%%%%%%%%%%%%%%%%%%%%%
%Computation of do

do=sqrt(Efs/Emp);

%Creation of the random Sensor Network
figure(1);
for i=1:1:n
    S(i).xd=rand(1,1)*xm;%×ø±ê
    XR(i)=S(i).xd;
    S(i).yd=rand(1,1)*ym;
    YR(i)=S(i).yd;
    S(i).G=0;
    %initially there are no cluster heads only nodes
    S(i).type='N';%ÆÕÍ¨½Úµã
    S(i).E=Eo;
    S(i).ENERGY=0;
    % hold on;
end

S(n+1).xd=sink.x;
S(n+1).yd=sink.y;

%First Iteration
figure(1);

%counter for CHs
countCHs=0;
%counter for CHs per round
rcountCHs=0;
cluster=1;

countCHs;
rcountCHs=rcountCHs+countCHs;
flag_first_dead=0;

for r=0:1:rmax %Ö÷Ñ»·,Ã¿´Î1ÂÖ
  r
  %Operation for epoch
  if(mod(r, round(1/p))==0)
     for i=1:1:n
        S(i).G=0;
        S(i).cl=0;
     end
  end
```

```
hold off;

%Number of dead nodes
dead=0;

%counter for bit transmitted to Bases Station and to Cluster Heads
packets_TO_BS=0;
packets_TO_CH=0;
%counter for bit transmitted to Bases Station and to Cluster Heads per round
PACKETS_TO_CH(r+1)=0;
PACKETS_TO_BS(r+1)=0;

figure(1);

for i=1:1:n
    %checking if there is a dead node
     if (S(i).E<=0)
       dead=dead+1;
     end

     if (S(i).E>0)
        S(i).type='N';
     end
end

if (dead == n)%½ÚµãÈ«²¿ËÀÍöÍË³öÑ»·
   break;
end

STATISTICS(r+1).DEAD=dead;
DEAD(r+1)=dead;

%When the first node dies
if (dead==1)
    if(flag_first_dead==0)
        first_dead=r
        flag_first_dead=1;
    end
end

countCHs=0;
cluster=1;
for i=1:1:n
   if(S(i).E>0)
     temp_rand=rand;
     if ((S(i).G)<=0) %Èç¹û¸Ã½ÚµãÔÚºòÑ¡¼¯ºÏÖÐ
        %Election of Cluster Heads
        if(temp_rand <=(p/(1-p*mod(r,round(1/p)))))
            countCHs = countCHs+1;

            S(i).type = 'C';
            S(i).G = round(1/p)-1;
            C(cluster).xd = S(i).xd;
            C(cluster).yd = S(i).yd;

            distance=sqrt((S(i).xd-(S(n+1).xd))^2         +          (S(i).yd-
(S(n+1).yd))^2);%µ½sinkµÄ¾àÀë

            C(cluster).distance = distance;
            C(cluster).id = i;
            X(cluster)=S(i).xd;
            Y(cluster)=S(i).yd;
            cluster=cluster+1;
            %¹ã²¥×Ô³ÉÎª´ØÍ·
            distanceBroad = sqrt(xm*xm+ym*ym);
            if (distanceBroad >=do)
                S(i).E         =         S(i).E-(ETX*ctrPacketLength         +
Emp*ctrPacketLength*(distanceBroad*distanceBroad*distanceBroad*distanceBroad)
);%¹ã²¥×Ô³ÉÎª´ØÍ·
            else
                S(i).E         =         S(i).E-(ETX*ctrPacketLength         +
Efs*ctrPacketLength*(distanceBroad*distanceBroad));
            end
            %Calculation of Energy dissipated ´ØÍ·×Ô¼º·¢ËÍÊý¾Ý°üÄÜÁ¿Ïû°Ä
            distance;
            if(distance>=do)
                 S(i).E           =           S(i).E-((ETX+EDA)*packetLength+
Emp*packetLength*(distance*distance*distance*distance ));
            else
                 S(i).E           =           S(i).E-((ETX+EDA)*packetLength+
Efs*packetLength*(distance*distance));
            end
            packets_TO_BS = packets_TO_BS+1;
            PACKETS_TO_BS(r+1) = packets_TO_BS;
        end
     end
   end
end

STATISTICS(r+1).CLUSTERHEADS                          =                     cluster-
1;%Í³¼ÆµÚrÂÖ´ØÍ·ÊýÄ¿,rÊÇ´Ó0¿ªÊ¼µÄ,ËùÒÔ¼Ó1;cluster×î°óÒª-1,ÊÇÉÏÃæµÄÑ-»·¶à¼ÓÁË1
CLUSTERHS(r+1)= cluster-1;

%Election of Associated Cluster Head for Normal Nodes
for i=1:1:n
   if (S(i).type=='N' && S(i).E>0) %ÆÕÍ¨½Úµã
    %   min_dis   =   sqrt(   (S(i).xd-S(n+1).xd)^2   +   (S(i).yd-S(n+1).yd)^2
);%Ä¬ÈÏ¾àÀëÊÇµ½sinkµÄ¾àÀë
     min_dis = INFINITY;
     if(cluster-1>=1)%Èç¹ûÓÐ´ØÍ·´æÔÚ
         min_dis_cluster = 1;
         %¼ÓÈë×î½üµÄ´ØÍ·
         for c = 1:1:cluster-1 %´ØÍ·ÊýÁ¿Ò»¹²ÊÇcluster-1
            %temp  =  min(min_dis,sqrt(  (S(i).xd  -  C(c).xd)^2  +  (S(i).yd  -
C(c).yd)^2 ) );
            temp = sqrt((S(i).xd - C(c).xd)^2 + (S(i).yd - C(c).yd)^2);
            if (temp<min_dis)
                min_dis = temp;
                min_dis_cluster = c;
            end
            %½ÓÊÕ´ØÍ··¢À´µÄ¹ã²¥µÄÏû°Ä
            S(i).E = S(i).E - ETX * ctrPacketLength;
         end

         %Energy dissipated by associated Cluster
HeadÆÕÍ¨½Úµã·¢ËÍÊý¾Ý°üµ½´ØÍ·Ïû°Ä,°Í¼ÓÈëÏûÏ¢
         min_dis;
         if (min_dis > do)
             S(i).E = S(i).E - (ETX*(ctrPacketLength) + Emp *
ctrPacketLength*( min_dis * min_dis * min_dis * min_dis));
%Ïò´ØÍ··¢ËÍ¼ÓÈë¿ØÖÆÏûÏ¢
             S(i).E = S(i).E - (ETX*(packetLength) + Emp*packetLength*(
min_dis * min_dis * min_dis * min_dis)); %Ïò´ØÍ·Êý¾Ý°ü
         else
            S(i).E = S(i).E -(ETX*(ctrPacketLength) + Efs*ctrPacketLength*(
min_dis * min_dis)); %Ïò´ØÍ··¢ËÍ¼ÓÈë¿ØÖÆÏûÏ¢
            S(i).E = S(i).E -(ETX*(packetLength) + Efs*packetLength*( min_dis
* min_dis)); %Ïò´ØÍ·Êý¾Ý°ü
         end
         S(i).E = S(i).E - ETX*(ctrPacketLength);  %½ÓÊÕ´ØÍ·È·ÈÏ¼ÓÈë¿ØÖÆÏûÏ¢

         %Energy dissipated
%´ØÍ·½ÓÊÕ´Ø³ÉÔ±Êý¾Ý°üÏû°ÄÄÜÁ¿,½ÓÊÕ¼ÓÈëÏûÏ¢°Í°ÍÈ·ÈÏ¼ÓÈëÏûÏ¢
         if(min_dis > 0)
            S(C(min_dis_cluster).id).E = S(C(min_dis_cluster).id).E - ((ERX +
EDA)*packetLength ); %½ÓÊÜ´Ø³ÉÔ±·¢À´µÄÊý¾Ý°ü
            S(C(min_dis_cluster).id).E = S(C(min_dis_cluster).id).E - ERX
*ctrPacketLength ; %½ÓÊÕ¼ÓÈëÏûÏ¢
            if (min_dis > do)%´ØÍ·Ïò´Ø³ÉÔ±·¢ËÍÈ·ÈÏ¼ÓÈëµÄÏûÏ¢
                S(C(min_dis_cluster).id).E = S(C(min_dis_cluster).id).E - (
ETX*(ctrPacketLength) + Emp * ctrPacketLength*( min_dis * min_dis * min_dis *
min_dis));
            else
                S(C(min_dis_cluster).id).E = S(C(min_dis_cluster).id).E - (
ETX*(ctrPacketLength) + Efs * ctrPacketLength*( min_dis * min_dis));
            end
           PACKETS_TO_CH(r+1) = n - dead - cluster + 1;
%ËùÓÐµÄ·ÇËÀÍöµÄÆÕÍ¨½Úµã¶¼·¢ËÍÊý¾Ý°ü
         end

         S(i).min_dis = min_dis;
         S(i).min_dis_cluster = min_dis_cluster;

     end
  end
end
%hold on;

countCHs;
rcountCHs = rcountCHs + countCHs;
```

```
end

x=1:1:r;
y=1:1:r;
%z=1:1:r;

for i=1:1:r;
    x(i)=i;
    y(i) = n - STATISTICS(i).DEAD;
    %z(i)=CLUSTERHS(i);
end
%plot(x,y,'r',x,z,'b');
plot(x,y,'r');
hold on;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%                                 STATISTICS
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%
%  DEAD  : a rmax x 1 array of number of dead nodes/round
%  DEAD_A : a rmax x 1 array of number of dead Advanced nodes/round
%  DEAD_N : a rmax x 1 array of number of dead Normal nodes/round
%  CLUSTERHS : a rmax x 1 array of number of Cluster Heads/round
%   PACKETS_TO_BS  :  a  rmax  x  1  array  of  number  packets  send  to  Base
Station/round
%   PACKETS_TO_CH  :  a  rmax  x  1  array  of  number  of  packets  send  to
ClusterHeads/round
%  first_dead: the round where the first node died
%
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%
%±ê×¼Êý¾Ý£°sink(50,175) ,ctrPacketLength=200,packetLength=4000,Eo=2J.
```

Appendix B:

Gr Theory:

```
% Test program for GrTheory functions
% Author: Sergii Iglin
% e-mail: siglin@yandex.ru
% personal page: http://iglin.exponenta.ru

clear all

% select test:
% ntest=1 - grBase
% ntest=2 - grCoBase
% ntest=3 - grCoCycleBasis
% ntest=4 - grColEdge
% ntest=5 - grColVer
% ntest=6 - grComp
% ntest=7 - grCycleBasis
% ntest=8 - grDecOrd
% ntest=9 - grDistances
% ntest=10 - grEccentricity
% ntest=11 - grIsEulerian
% ntest=12 - grIsomorph
% ntest=13 - grMaxComSu
% ntest=14 - grMaxFlows
% ntest=15 - grMaxMatch
% ntest=16 - grMaxStabSet
% ntest=17 - grMinAbsEdgeSet
% ntest=18 - grMinAbsVerSet
% ntest=19 - grMinCutSet
% ntest=20 - grMinEdgeCover
% ntest=21 - grMinSpanTree
% ntest=22 - grMinVerCover
% ntest=23 - grPERT
% ntest=24 - grPlot
% ntest=25 - grShortPath
% ntest=26 - grShortVerPath
% ntest=27 - grTranClos
% ntest=28 - grTravSale

ntest=26;
```

```
switch ntest % selected test
  case 1, % grBase test
    disp('The grBase test')
    V=[0 4;4 4;0 0;4 0;8 4;8 0;12 4;12 0;...
      -2 8;-4 4;0 -4;-4 -4;-4 0];
    E=[1 2;3 4;2 4;2 5;4 6;5 6;5 7;6 8;8 7;...
       1 9;9 10;10 1;3 11;11 12;12 13;13 3;3 12;13 11];
    grPlot(V,E,'d','%d','');
    title('\bfThe initial digraph')
    BG=grBase(E);
    disp('The bases of digraph:')
    disp(' N    vertexes')
    for k1=1:size(BG,1),
      fprintf('%2.0f    ',k1)
      fprintf('%d  ',BG(k1,:))
      fprintf('\n')
    end
  case 2, % grCoBase test
    disp('The grCoBase test')
    V=[0 4;4 4;0 0;4 0;8 4;8 0;12 4;12 0;...
      -2 8;-4 4;0 -4;-4 -4;-4 0];
    E=[2 1;3 4;2 4;2 5;4 6;5 6;5 7;6 8;8 7;...
       1 9;9 10;10 1;3 11;11 12;12 13;13 3;3 12;13 11];
    grPlot(V,E,'d','%d','');
    title('\bfThe initial digraph')
    CBG=grCoBase(E);
    disp('The contrabasis of digraph:')
    disp(' N    vertexes')
    for k1=1:size(CBG,1),
      fprintf('%2.0f    ',k1)
      fprintf('%d  ',CBG(k1,:))
      fprintf('\n')
    end
  case 3, % grCoCycleBasis test
    disp('The grCoCycleBasis test')
    V=[0 0;1 1;1 0;1 -1;2 1;2 0;2 -1;3 1;...
       3 0;3 -1;4 0]; % vertexes coordinates
    E=[1 2;1 3;1 4;2 3;3 4;2 5;2 6;3 6;3 7;4 7;5 6;6 7;...
       5 8;6 8;6 9;7 9;7 10;8 9;9 10;8 11;9 11;10 11]; % edges
    E=[E [1:size(E,1)]']; % edges with numbers
    grPlot(V,E,'g','','%d'); % the initial graph
    title('\bfThe initial graph')
    CoCycles=grCoCycleBasis(E); % all independences cut-sets
    for k1=1:size(CoCycles,2),
      grPlot(V,E(find(~CoCycles(:,k1)),:),'g','','%d'); % one cocycle
      title(['\bfCocycle N' num2str(k1)]);
    end
  case 4, % grColEdge test
    disp('The grColEdge test')
    V=[0 0;1 1;1 0;1 -1;2 1;2 0;2 -1;3 1;...
       3 0;3 -1;4 0]; % vertexes coordinates
    E=[1 2;1 3;1 4;2 3;3 4;2 5;2 6;3 6;3 7;4 7;5 6;6 7;...
       5 8;6 8;6 9;7 9;7 10;8 9;9 10;8 11;9 11;10 11]; % edges
    grPlot(V,E,'g','','%d'); % the initial graph
    title('\bfThe initial graph')
    mCol=grColEdge(E); % the color problem for edges
    fprintf('The colors of edges\n N edge    N col\n');
    fprintf('  %2.0f        %2.0f\n',[1:length(mCol);mCol']);
    grPlot(V,[E,mCol],'g','','%d'); % plot the colored graph
    title('\bfThe graph with colored edges')
  case 5, % grColVer test
    disp('The grColVer test')
    t=[0:4]';
    V=[[5*sin(2*pi*t/5) 5*cos(2*pi*t/5)];...
       [4*sin(2*pi*(t-0.5)/5) 4*cos(2*pi*(t-0.5)/5)];...
       [2*sin(2*pi*(t-0.5)/5) 2*cos(2*pi*(t-0.5)/5)];[0 0]];
    E=[1 7;7 2;2 8;8 3;3 9;9 4;4 10;10 5;5 6;6 1;...
      1 10;2 6;3 7;4 8;5 9;1 12;2 13;3 14;4 15;5 11;...
      6 14;7 15;8 11;9 12;10 13;...
      11 12;12 13;13 14;14 15;15 11;1 16;2 16;3 16;4 16;5 16];
    grPlot(V,E,'g','%d',''); % the initial graph
    title('\bfThe initial graph')
    nCol=grColVer(E); % the color problem for vertexes
    fprintf('The colors of vertexes\n N ver     N col\n');
    fprintf('  %2.0f        %2.0f\n',[1:length(nCol);nCol']);
    grPlot([V,nCol],E,'g','%d',''); % plot the colored graph
    title('\bfThe graph with colored vertexes')
  case 6, % grComp test
    disp('The grComp test')
    V=[0 4;4 4;0 0;4 0;8 4;8 0;12 4;12 0;...
      -2 8;-4 4;0 -4;-4 -4;-4 0];
    E=[1 2;3 4;2 4;2 5;4 6;5 6;5 7;6 8;8 7;...
       1 9;9 10;10 1;3 11;11 12;12 13;13 3;3 12;13 11];
    ncV=grComp(E);
    grPlot([V ncV],E,'g','%d','');
    title(['\bfThis graph have ' num2str(max(ncV)) ' component(s)'])
    E=[2 4;2 5;4 6;5 6;5 7;6 8;8 7;...
       1 9;9 10;10 1;3 11;11 12;12 13;13 3;3 12;13 11];
    ncV=grComp(E);
    grPlot([V ncV],E,'g','%d','');
    title(['\bfThis graph have ' num2str(max(ncV)) ' component(s)'])
    E=[2 4;2 5;4 6;5 6;5 7;6 8;8 7;...
       1 9;9 10;10 1;3 11;11 12;3 12];
    ncV=grComp(E,size(V,1));
    grPlot([V ncV],E,'g','%d','');
    title(['\bfThis graph have ' num2str(max(ncV)) ' component(s)'])
  case 7, % grCycleBasis test
    disp('The grCycleBasis test')
    V=[0 0;1 1;1 0;1 -1;2 1;2 0;2 -1;3 1;...
       3 0;3 -1;4 0]; % vertexes coordinates
    E=[1 2;1 3;1 4;2 3;3 4;2 5;2 6;3 6;3 7;4 7;5 6;6 7;...
       5 8;6 8;6 9;7 9;7 10;8 9;9 10;8 11;9 11;10 11]; % edges
    grPlot(V,E,'g','%d',''); % the initial graph
    title('\bfThe initial graph')
    Cycles=grCycleBasis(E); % all independences cycles
    for k1=1:size(Cycles,2),
      grPlot(V,E(find(Cycles(:,k1)),:),'g','%d',''); % one cycle
      title(['\bfCycle N' num2str(k1)]);
    end
  case 8, % grDecOrd test
```

```
    disp('The grDecOrd test')
    V=[0 4;1 4;2 4;3 4;4 4;0 3;1 3;2 3;3 3;4 3;...
       0 2;1 2;2 2;3 2;4 2;0 1;1 1;2 1;3 1;4 1;...
       0 0;1 0;2 0;3 0;4 0];
    E=[1 2;3 2;4 3;5 4;6 1;2 7;8 2;3 8;9 4;9 5;10 5;7 6;8 7;...
       8 9;10 9;11 6;7 12;13 8;14 9;15 10;12 11;13 12;13 14;...
       14 13;15 14;16 11;12 17;13 18;20 15;17 16;17 18;18 17;...
       19 18;19 20;21 16;17 22;18 22;22 18;18 23;19 24;20 25;...
       21 22;22 21;23 24;24 23;24 25];
    grPlot(V,E,'d');
    title('\bfThe initial digraph')
    [Dec,Ord]=grDecOrd(E); % solution
    disp('The classes of mutually connected vertexes:')
    disp(' N    vertexes')
    for k1=1:size(Dec,2),
      fprintf('%2.0f    ',k1)
      fprintf('%d  ',find(Dec(:,k1)))
      fprintf('\n')
    end
    fprintf('The partial ordering of the classes:\n  ')
    fprintf('%3.0f',1:size(Ord,2))
    fprintf('\n')
    for k1=1:size(Ord,1), % the matrix of partial ordering
      fprintf('%2.0f ',k1)
      fprintf(' %1.0f ',Ord(k1,:))
      fprintf('\n')
    end
    V1=V;
    for k1=1:size(Dec,2),
      V1(find(Dec(:,k1)),3)=k1; % weight = number of the class
    end
    grPlot(V1,E,'d','%d','');
    title('\bfThe classes of mutually connected vertexes')
  case 9, % grDispances test
    disp('The grDispances test')
    V=[0 4;1 4;2 4;3 4;4 4;0 3;1 3;2 3;3 3;4 3;...
       0 2;1 2;2 2;3 2;4 2;0 1;1 1;2 1;3 1;4 1;...
       0 0;1 0;2 0;3 0;4 0];
    E=[1 2 1;3 2 2;4 3 3;5 4 4;6 1 5;2 7 6;8 2 7;3 8 1;...
       9 4 9;9 5 8;10 5 7;7 6 6;8 7 5;8 9 4;10 9 3;11 6 2;...
       7 12 1;13 8 2;14 9 3;15 10 4;12 11 5;13 12 6;13 14 7;...
       14 13 8;5 5 10;15 14 9;16 11 8;12 17 7;13 18 6;...
       20 15 5;17 16 4;17 18 3;18 17 2;19 18 1;19 20 2;...
       5 5 8; 21 16 3;17 22 4;18 22 5;22 18 6;18 23 7;...
       19 24 8;20 25 9;21 22 8;22 21 7;23 24 6;10 10 8;...
       24 23 5;24 25 4];
    s=1; % one vertex
    t=25; % other vertex
    grPlot(V,E); % the initial graph
    title('\bfThe initial graph with weighed edges')
    [dSP,sp]=grDistances(E(:,1:2),s,t); % the nonweighted task
    fprintf('The steps number between all vertexes:\n   ')
    fprintf('%4.0f',1:size(dSP,2))
    fprintf('\n')
    for k1=1:size(dSP,1), % the matrix of distances
```

```
      fprintf('%3.0f ',k1)
      fprintf(' %2.0f ',dSP(k1,:))
      fprintf('\n')
    end
    grPlot(V(:,1:2),[sp(1:end-1);sp(2:end)]','d','%d','')
    title(['\bfThe shortest way between vertex ' ...
      num2str(s) ' and vertex ' num2str(t)])
    [dSP,sp]=grDistances(E,1,25); % the weighted task
    fprintf('The distances between all vertexes:\n   ')
    fprintf('%4.0f',1:size(dSP,2))
    fprintf('\n')
    for k1=1:size(dSP,1), % the matrix of distances
      fprintf('%3.0f ',k1)
      fprintf(' %2.0f ',dSP(k1,:))
      fprintf('\n')
    end
    grPlot(V(:,1:2),[sp(1:end-1);sp(2:end)]','d','%d','')
    title(['\bfThe way with minimal weight between vertex ' ...
      num2str(s) ' and vertex ' num2str(t)])
  case 10, % grEccentricity test
    disp('The grEccentricity test')
    V=[0 4;1 4;2 4;3 4;4 4;0 3;1 3;2 3;3 3;4 3;...
       0 2;1 2;2 2;3 2;4 2;0 1;1 1;2 1;3 1;4 1;...
       0 0;1 0;2 0;3 0;4 0];
    E=[1 2 1;3 2 2;4 3 3;5 4 4;6 1 5;2 7 6;8 2 7;3 8 1;...
       9 4 9;9 5 8;10 5 7;7 6 6;8 7 5;8 9 4;10 9 3;11 6 2;...
       7 12 1;13 8 2;14 9 3;15 10 4;12 11 5;13 12 6;13 14 7;...
       14 13 8;5 5 10;15 14 9;16 11 8;12 17 7;13 18 6;...
       20 15 5;17 16 4;17 18 3;18 17 2;19 18 1;19 20 2;...
       5 5 8; 21 16 3;17 22 4;18 22 5;22 18 6;18 23 7;...
       19 24 8;20 25 9;21 22 8;22 21 7;23 24 6;10 10 8;...
       24 23 5;24 25 4];
    grPlot(V,E); % the initial graph
    title('\bfThe initial graph with weighed edges')
    [Ec,Rad,Diam,Cv,Pv]=grEccentricity(E(:,1:2)); % the nonweighted task
    fprintf('The nonweighted eccentricities of all vertexes:\n N ver  Ecc\n')
    fprintf('%4.0f  %4.0f\n',[1:size(V,1);Ec])
    fprintf('The radius of graph Rad=%d\n',Rad)
    fprintf('The diameter of graph Diam=%d\n',Diam)
    fprintf('The center vertexes is:')
    fprintf('  %d',Cv)
    fprintf('\nThe periphery vertexes is:')
    fprintf('  %d',Pv)
    fprintf('\n')
    [Ec,Rad,Diam,Cv,Pv]=grEccentricity(E); % the weighted task
    fprintf('The weighted eccentricities of all vertexes:\n N ver  Ecc\n')
    fprintf('%4.0f  %4.0f\n',[1:size(V,1);Ec])
    fprintf('The radius of graph Rad=%d\n',Rad)
    fprintf('The diameter of graph Diam=%d\n',Diam)
    fprintf('The center vertexes is:')
    fprintf('  %d',Cv)
    fprintf('\nThe periphery vertexes is:')
    fprintf('  %d',Pv)
    fprintf('\n')
  case 11, % grIsEulerian test
```

```
disp('The grIsEulerian test')
V=[0 4;1 4;2 4;3 4;4 4;0 3;1 3;2 3;3 3;4 3;...
   0 2;1 2;2 2;3 2;4 2;0 1;1 1;2 1;3 1;4 1;...
   0 0;1 0;2 0;3 0;4 0];
E=[1 2;3 2;4 3;5 4;6 1;2 7;8 2;3 8;9 4;9 5;10 5;7 6;8 7;...
   8 9;10 9;11 6;7 12;13 8;14 9;15 10;12 11;13 12;13 14;...
   14 13;15 14;16 11;12 17;13 18;20 15;17 16;17 18;18 17;...
   19 18;19 20;21 16;17 22;18 22;22 18;18 23;19 24;20 25;...
   21 22;22 21;23 24;24 23;24 25];
[eu,cEu]=grIsEulerian(E);
switch eu,
  case 1,
    st='';
    E=[E(cEu,1:2), [1:size(E,1)]'];
  case 0.5,
    st='semi-';
    E=[E(cEu,1:2), [1:size(E,1)]'];
  otherwise,
    st='not ';
    E=[E(:,1:2), [1:size(E,1)]'];
end
grPlot(V,E,'g','','%d');
title(['\bf This graph is ' st 'Eulerian'])
V=[0 4;1 4;2 4;3 3.7;4 4;0 3;1 3.3;2 3.3;3 3;4 3;...
   0 2;1 2;2 2;3 2;4 2;0 1;1 1;2 1;3 1;4 1;...
   0 0;1 0;2 0;3 0;4 0];
E=[1 2;3 2;4 3;5 4;3 5;6 1;2 7;8 2;3 8;9 4;9 5;...
   10 4;10 5;7 6;8 7;6 9;20 15;11 16;21 18;19 23;...
   8 9;10 9;11 6;7 12;14 9;15 10;12 11;13 12;13 14;...
   14 13;15 14;16 11;12 17;13 18;20 15;17 16;17 18;...
   19 18;19 20;21 16;17 22;18 22;18 23;19 24;20 25;...
   21 22;22 21;23 24;24 23;24 25];
[eu,cEu]=grIsEulerian(E);
switch eu,
  case 1,
    st='';
    E=[E(cEu,1:2), [1:size(E,1)]'];
  case 0.5,
    st='semi-';
    E=[E(cEu,1:2), [1:size(E,1)]'];
  otherwise,
    st='not ';
    E=[E(:,1:2), [1:size(E,1)]'];
end
grPlot(V,[E,[1:size(E,1)]'],'g','');
title(['\bf This graph is ' st 'Eulerian'])
V=[0 4;1 4;2 4;3 3.7;4 4;0 3;1 3.3;2 3.3;3 3;4 3;...
   0 2;1 2;2 2;3 2;4 2;0 1;1 1;2 1;3 1;4 1;...
   0 0;1 0;2 0;3 0;4 0];
E=[1 2;3 2;4 3;5 4;3 5;6 1;2 7;8 2;3 8;9 4;9 5;...
   10 4;10 5;7 6;8 7;6 9;20 15;11 16;21 18;19 23;...
   8 9;10 9;11 6;7 12;14 9;15 10;12 11;13 12;13 14;...
   14 13;15 14;16 11;12 17;13 18;20 15;17 16;17 18;...
   19 18;19 20;21 16;17 22;18 22;18 23;19 24;20 25;...
   21 22;22 21;23 24;24 25];
```

```
    [eu,cEu]=grIsEulerian(E);
    switch eu,
      case 1,
        st='';
        E=[E(cEu,1:2), [1:size(E,1)]'];
      case 0.5,
        st='semi-';
        E=[E(cEu,1:2), [1:size(E,1)]'];
      otherwise,
        st='not ';
        E=[E(:,1:2), [1:size(E,1)]'];
    end
    grPlot(V,[E,[1:size(E,1)]'],'g','');
    title(['\bf This graph is ' st 'Eulerian'])
  case 12, % grIsomorph test
    disp('The grIsomorph test')
    V1=[0 0;1 0;2 0;0 1;1 1;2 1];
    E1=[1 4;1 5;1 6;2 4;2 5;2 6;3 4;3 5;3 6];
    V2=[cos([1:6]*pi/3);sin([1:6]*pi/3)]';
    E2=[1 2;2 3;3 4;4 5;5 6;6 1;1 4;2 5;3 6];
    grPlot(V1,E1,'g','%d','');
    title('\bfThe graph 1')
    grPlot(V2,E2,'g','%d','');
    title('\bfThe graph 2')
    [IsIsomorph,Permut]=grIsomorph(E1,E2);
    if IsIsomorph,
      disp('Their graphs are isomorphic:')
      fprintf('Number of vertexes:\n  V1   V2\n')
      fprintf(' %3.0f  %3.0f\n',[[1:max(E1(:))];Permut'])
    else
      disp('Their graphs are nonisomorphic')
    end;
  case 13, % grMaxComSu test
    disp('The grMaxComSu test')
    V=[0 0 2;1 1 3;1 0 3;1 -1 4;2 1 1;2 0 2;2 -1 3;3 1 4;...
       3 0 5;3 -1 1;4 0 5]; % vertexes coordinates and weights
    E=[1 2;1 3;1 4;2 3;3 4;2 5;2 6;3 6;3 7;4 7;5 6;6 7;...
       5 8;6 8;6 9;7 9;7 10;8 9;9 10;8 11;9 11;10 11]; % edges
    grPlot(V(:,1:2),E,'g','%d',''); % the initial graph
    title('\bfThe initial graph')
    grPlot(V,E,'g','%d',''); % the initial graph
    title('\bfThe initial graph with weighed vertexes')
    nMS=grMaxComSu(E); % the maximal complete sugraph
    fprintf('Number of vertexes on the maximal complete sugraph = %d\n',...
      length(nMS));
    disp('In a maximal complete sugraph is the vertexes with numbers:');
    fprintf('%d  ',nMS);
    fprintf('\nThe total weight = %d\n',sum(V(nMS,3)));
    nMS=grMaxComSu(E,V(:,3)); % the weightd maximal complete sugraph
    fprintf(['Number of vertexes on the weighed maximal complete sugraph '...
      '= %d\n'],length(nMS));
    disp('In a weighed maximal complete sugraph is the vertexes with
numbers:');
    fprintf('%d  ',nMS);
    fprintf('\nThe total weight = %d\n',sum(V(nMS,3)));
```

```
case 14, % grMaxFlows test
  disp('The grMaxFlows test')
  V=[0 0;1 1;1 0;1 -1;2 1;2 0;2 -1;3 1;...
     3 0;3 -1;4 0]; % vertexes coordinates
  E=[1 2 5;1 3 5;1 4 5;2 3 2;3 4 2;2 5 3;...
     2 6 2;3 6 5;3 7 2;4 7 3;5 6 1;6 7 1;...
     5 8 5;6 8 2;6 9 3;7 9 2;7 10 3;8 9 2;...
     9 10 2;8 11 5;9 11 4;10 11 4]; % arrows and weights
  s=1; % the network source
  t=11; % the network sink
  fprintf('The source of the net s=%d\nThe sink of the net t=%d\n',s,t)
  grPlot(V,E,'d','','%d'); % the initial digraph
  title('\bfThe digraph of the net')
  [v,mf]=grMaxFlows(E,s,t); % the maximal flow
  disp('The solution of the maximal flows problem')
  disp('  N arrow       flow')
  fprintf('   %2.0f      %12.8f\n',[[1:length(v)];v'])
  fprintf('The maximal flow =%12.8f\n',mf)
  grPlot(V,[E(:,1:2),v],'d','','%6.4f'); % plot the digraph
  title('\bfThe flows on the arrows')
case 15, % grMaxMatch test
  disp('The grMaxMatch test')
  V=[0 0;1 1;1 0;1 -1;2 1;2 0;2 -1;3 1;...
     3 0;3 -1;4 0]; % vertexes coordinates
  E=[1 2 5;1 3 5;1 4 5;2 3 2;3 4 2;2 5 3;...
     2 6 2;3 6 5;3 7 2;4 7 3;5 6 1;6 7 1;...
     5 8 5;6 8 2;6 9 3;7 9 2;7 10 3;8 9 2;...
     9 10 2;8 11 5;9 11 4;10 11 4]; % arrows and weights
  grPlot(V,E,'g','','%d'); % the initial graph
  title('\bfThe initial graph with weighed edges')
  nMM=grMaxMatch(E(:,1:2)); % the maximal matching
  fprintf('Number of edges on the maximal matching = %d\n',...
    length(nMM));
  disp('In a maximal matching is the edges with numbers:');
  fprintf('%d  ',nMM);
  fprintf('\nThe total weight = %d\n',sum(E(nMM,3)));
  grPlot(V,E(nMM,:),'g','','%d'); % the maximal matching
  title('\bfThe maximal matching')
  nMM=grMaxMatch(E); % the weighed maximal matching
  fprintf('Number of edges on the weighed maximal matching = %d\n',...
    length(nMM));
  disp('In a weighed maximal matching is the edges with numbers:');
  fprintf('%d  ',nMM);
  fprintf('\nThe total weight = %d\n',sum(E(nMM,3)));
  grPlot(V,E(nMM,:),'g','','%d'); % the weighed maximal matching
  title('\bfThe weighed maximal matching')
case 16, % grMaxStabSet test
  disp('The grMaxStabSet test')
  V=[0 0 2;1 1 3;1 0 3;1 -1 4;2 1 1;2 0 2;2 -1 3;3 1 4;...
     3 0 5;3 -1 1;4 0 5]; % vertexes coordinates and weights
  E=[1 2;1 3;1 4;2 3;3 4;2 5;2 6;3 6;3 7;4 7;5 6;6 7;...
     5 8;6 8;6 9;7 9;7 10;8 9;9 10;8 11;9 11;10 11]; % edges
  grPlot(V(:,1:2),E,'g','%d',''); % the initial graph
  title('\bfThe initial graph')
  nMS=grMaxStabSet(E); % the maximal stable set
```

```
    fprintf('Number of vertexes on the maximal stable set = %d\n',...
      length(nMS));
    disp('In a maximal stable set is the vertexes with numbers:');
    fprintf('%d  ',nMS);
    fprintf('\nThe total weight = %d\n',sum(V(nMS,3)));
    grPlot(V,E,'g','%d',''); % the initial graph
    title('\bfThe initial graph with weighed vertexes')
    nMS=grMaxStabSet(E,V(:,3)); % the weightd maximal stable set
    fprintf(['Number of vertexes on the weighed maximal stable set '...
      '= %d\n'],length(nMS));
    disp('In a weighed maximal stable set is the vertexes with numbers:');
    fprintf('%d  ',nMS);
    fprintf('\nThe total weight = %d\n',sum(V(nMS,3)));
  case 17, % grMinAbsEdgeSet test
    disp('The grMinAbsEdgeSet test')
    V=[0 0;1 1;1 0;1 -1;2 1;2 0;2 -1;3 1;...
       3 0;3 -1;4 0]; % vertexes coordinates
    E=[1 2 5;1 3 5;1 4 5;2 3 2;3 4 2;2 5 3;...
       2 6 2;3 6 5;3 7 2;4 7 3;5 6 1;6 7 1;...
       5 8 5;6 8 2;6 9 3;7 9 2;7 10 3;8 9 4;...
       9 10 5;8 11 5;9 11 4;10 11 4]; % arrows and weights
    grPlot(V,E,'g','','%d'); % the initial graph
    title('\bfThe initial graph with weighed edges')
    nMS=grMinAbsEdgeSet(E(:,1:2)); % the minimal absorbant set of edges
    fprintf('Number of edges on the minimal absorbant set = %d\n',...
      length(nMS));
    disp('In a minimal absorbant set is the edges with numbers:');
    fprintf('%d  ',nMS);
    fprintf('\nThe total weight = %d\n',sum(E(nMS,3)));
    grPlot(V,E(nMS,:),'g','','%d'); % the minimal absorbant set of edges
    title('\bfThe minimal absorbant set of edges')
    nMS=grMinAbsEdgeSet(E); % the minimal weighed absorbant set of edges
    fprintf('Number of edges on the minimal weighed absorbant set = %d\n',...
      length(nMS));
    disp('In a minimal weighed absorbant set is the edges with numbers:');
    fprintf('%d  ',nMS);
    fprintf('\nThe total weight = %d\n',sum(E(nMS,3)));
    grPlot(V,E(nMS,:),'g','','%d');  %  the  minimal  weighed  absorbant  set  of
edges
    title('\bfThe minimal weighed absorbant set of edges')
  case 18, % grMinAbsVerSet test
    disp('The grMinAbsVerSet test')
    V=[0 0 2;1 1 3;1 0 3;1 -1 4;2 1 1;2 0 2;2 -1 3;3 1 4;...
       3 0 5;3 -1 1;4 0 5]; % vertexes coordinates and weights
    E=[1 2;1 3;1 4;2 3;3 4;2 5;2 6;3 6;3 7;4 7;5 6;6 7;...
       5 8;6 8;6 9;7 9;7 10;8 9;9 10;8 11;9 11;10 11]; % edges
    grPlot(V(:,1:2),E,'g','%d',''); % the initial graph
    title('\bfThe initial graph')
    grPlot(V,E,'g','%d',''); % the initial graph
    title('\bfThe initial graph with weighed vertexes')
    nMS=grMinAbsVerSet(E); % the minimal absorbant set of vertexes
    fprintf('Number of vertexes on the minimal absorbant set = %d\n',...
      length(nMS));
    disp('In a minimal absorbant set is the vertexes with numbers:');
    fprintf('%d  ',nMS);
    fprintf('\nThe total weight = %d\n',sum(V(nMS,3)));
    nMS=grMinAbsVerSet(E,V(:,3));  %  the  weightd  minimal  absorbant  set  of
vertexes
    fprintf(['Number of vertexes on the weighed minimal absorbant set '...
      '= %d\n'],length(nMS));
    disp('In a weighed minimal absorbant set is the vertexes with numbers:');
    fprintf('%d  ',nMS);
    fprintf('\nThe total weight = %d\n',sum(V(nMS,3)));
  case 19, % grMinCutSet test
    disp('The grMinCutSet test')
    V=[0 0;1 1;1 0;1 -1;2 1;2 0;2 -1;3 1;...
       3 0;3 -1;4 0]; % vertexes coordinates
    E=[1 2 5;1 3 5;1 4 5;2 3 2;3 4 2;2 5 3;...
       2 6 2;3 6 5;3 7 2;4 7 3;5 6 1;6 7 1;...
       5 8 5;6 8 2;6 9 3;7 9 2;7 10 3;8 9 2;...
       9 10 2;8 11 5;9 11 4;10 11 4]; % arrows and weights
    s=1; % the network source
    t=11; % the network sink
    fprintf('The source of the net s=%d\nThe sink of the net t=%d\n',s,t)
    grPlot(V,E,'d','','%d'); % the initial digraph
    title('\bfThe digraph of the net')
    [nMCS,mf]=grMinCutSet(E,s,t); % the minimal cut-set
    fprintf('The first minimal cut-set include arrows:');
    fprintf('  %d',nMCS);
    fprintf(['\nThe maximal flow through '...
      'each minimal cut-set = %12.6f\n'],mf)
    grPlot(V,E(setdiff(1:size(E,1),nMCS),:),'d','','%d');
    title('\bfThe digraph without first minimal cut-set')
  case 20, % grMinEdgeCover test
    disp('The grMinEdgeCover test')
    V=[0 0;1 1;1 0;1 -1;2 1;2 0;2 -1;3 1;...
       3 0;3 -1;4 0]; % vertexes coordinates and weights
    E=[1 2 5;1 3 5;1 4 5;2 3 2;3 4 2;2 5 3;2 6 2;3 6 5;...
       3 7 2;4 7 3;5 6 1;6 7 1;5 8 5;6 8 2;6 9 3;7 9 2;...
       7 10 3;8 9 2;9 10 2;8 11 5;9 11 4;10 11 4]; % edges and weights
    grPlot(V,E,'g',''); % the initial graph
    title('\bfThe initial graph with weighed edges')
    nMC=grMinEdgeCover(E(:,1:2)); % the minimal edge covering
    fprintf('Number of edges on the minimal edge covering = %d\n',...
      length(nMC));
    disp('In a minimal edge cover is the edges with numbers:');
    fprintf('%d  ',nMC);
    fprintf('\nThe total weight = %d\n',sum(E(nMC,3)));
    grPlot(V,E(nMC,:),'g',''); % the minimal edge covering
    title('\bfThe minimal edge covering')
    nMC=grMinEdgeCover(E); % the weighed minimal edge covering
    fprintf('Number of edges on the weighed minimal edge covering = %d\n',...
      length(nMC));
    disp('In a weighed minimal edge cover is the edges with numbers:');
    fprintf('%d  ',nMC);
    fprintf('\nThe total weight = %d\n',sum(E(nMC,3)));
    grPlot(V,E(nMC,:),'g',''); % the weighed minimal edge covering
    title('\bfThe weighed minimal edge covering')
  case 21, % grMinSpanTree test
    disp('The grMinSpanTree test')
    V=[0 4;1 4;2 4;3 4;4 4;0 3;1 3;2 3;3 3;4 3;...
       0 2;1 2;2 2;3 2;4 2;0 1;1 1;2 1;3 1;4 1;...
       0 0;1 0;2 0;3 0;4 0];
    E=[1 2 1;3 2 2;4 3 3;5 4 4;6 1 5;2 7 6;8 2 7;3 8 8;...
       9 4 9;9 5 8;10 5 7;7 6 6;8 7 5;8 9 4;10 9 3;11 6 2;...
       7 12 1;13 8 2;14 9 3;15 10 4;12 11 5;13 12 6;13 14 7;...
       14 13 8;5 5 10;15 14 9;16 11 8;12 17 7;13 18 6;...
       20 15 5;17 16 4;17 18 3;18 17 2;19 18 1;19 20 2;...
       5 5 8; 21 16 3;17 22 4;18 22 5;22 18 6;18 23 7;...
       19 24 8;20 25 9;21 22 8;22 21 7;23 24 6;10 10 8;...
       24 23 5;24 25 4];
    grPlot(V,E); % the initial graph
    title('\bfThe initial graph with weighed edges')
    nMST=grMinSpanTree(E(:,1:2)); % the spanning tree
    fprintf('Number of edges on the spanning tree = %d\n',length(nMST));
    fprintf('The total weight = %d\n',sum(E(nMST,3)));
    grPlot(V,E(nMST,:)); % the spanning tree
    title('\bfThe spanning tree')
    nMST=grMinSpanTree(E); % the minimal spanning tree
    fprintf('Number of edges on the minimal spanning tree = %d\n',...
      length(nMST));
    fprintf('The total weight = %d\n',sum(E(nMST,3)));
    grPlot(V,E(nMST,:)); % the minimal spanning tree
    title('\bfThe minimal spanning tree')
  case 22, % grMinVerCover test
    disp('The grMinVerCover test')
    V=[0 0 2;1 1 3;1 0 3;1 -1 4;2 1 1;2 0 2;2 -1 3;3 1 4;...
       3 0 7;3 -1 1;4 0 5]; % vertexes coordinates and weights
    E=[1 2;1 3;1 4;2 3;3 4;2 5;2 6;3 6;3 7;4 7;6 5;6 7;...
       5 8;6 8;6 9;7 9;7 10;8 9;9 10;8 11;9 11;10 11]; % edges
    grPlot(V,E,'g','%d',''); % the initial graph
    title('\bfThe initial graph with weighed vertexes')
    nMC=grMinVerCover(E); % the minimal vertex cover
    fprintf('Number of vertexes on the minimal vertex cover = %d\n',...
      length(nMC));
    disp('In a minimal vertex cover is the vertexes with numbers:');
    fprintf('%d  ',nMC);
    fprintf('\nThe total weight = %d\n',sum(V(nMC,3)));
    grPlot(V(nMC,:)); % the solution of the MinVerCover problem
    title('\bfThe minimal vertex cover')
    nMC=grMinVerCover(E,V(:,3)); % the weightd minimal vertex cover
    fprintf(['Number of vertexes on the weighed minimal vertex cover '...
      '= %d\n'],length(nMC));
    disp('In a weighed minimal vertex cover is the vertexes with numbers:');
    fprintf('%d  ',nMC);
    fprintf('\nThe total weight = %d\n',sum(V(nMC,3)));
    grPlot(V(nMC,:)); % the solution of the weighed MinVerCover problem
    title('\bfThe weighed minimal vertex cover')
  case 23, % grPERT test
    disp('The grPERT test')
    V=[1 1;0 0;1 0;1 -1;2 1;2 0;2 -1;3 1;...
       4 0;3 -1;3 0]; % events coordinates
    E=[2 1 5;2 3 5;2 4 5;1 3 2;3 4 2;1 5 3;...
       1 6 2;3 6 5;3 7 2;4 7 3;5 6 1;6 7 1;...
       5 8 5;6 8 2;6 11 3;7 11 2;7 10 3;8 11 2;...
       11 10 2;8 9 5;11 9 4;10 9 4]; % works and their times
    grPlot(V,E,'d','%d','%d');
    title('\bfThe schema of project')
    [CrP,Ts,Td]=grPERT(E);
    grPlot([V Ts'],[CrP(1:end-1);CrP(2:end)]','d','%d','');
    title('\bfThe critical path and start times for events')
    grPlot([V Ts'],[E(:,1:2) Td],'d','%d','%d')
    title('\bfThe start times for events and delay times for works')
  case 24, % grPlot test
    disp('The grPlot test')
    V=[0 0 2;1 1 3;1 0 3;1 -1 4;2 1 1;2 0 2;2 -1 3;3 1 4;...
       3 0 5;3 -1 1;4 0 5]; % vertexes coordinates and weights
    E=[1 2 5;1 1 2;1 1 5;2 2 3;1 3 5;1 4 5;2 3 2;3 4 2;2 5 3;2 6 2;3 6 5;3 7
2;...
       4 7 3;5 6 1;6 7 1;5 8 5;6 8 2;6 9 3;7 9 2;7 10 3;8 9 2;...
       9  10  2;8  11  5;9  11  4;10  11  4;1  2  8;1  3  4;1  3  5;1  3  6];  %  edges
(arrows) and weights
    grPlot(V(:,1:2),E,'d');
    title('\bfThe digraph with weighed multiple arrows and loops')
    grPlot(V,E(:,1:2),[],'%d','');
    title('\bfThe graph with weighed vertexes without edges numeration')
    grPlot(V(:,1:2));
    title('\bfThe disconnected graph')
    grPlot([],fullfact([5 5]),'d','','',0.8)
    title('\bfThe directed clique\rm \itK\rm_5')
  case 25, % grShortPath test
    disp('The grShortPath test')
    V=[0 0;1 1;1 0;1 -1;2 1;2 0;2 -1;3 1;...
       3 0;3 -1;4 0]; % vertexes coordinates
    E=[1 2 5;1 3 5;1 4 5;2 3 2;3 4 2;2 5 3;...
       2 6 2;3 6 5;3 7 2;4 7 3;5 6 1;6 7 1;...
       5 8 5;6 8 2;6 9 3;7 9 2;7 10 3;8 9 2;...
       9 10 2;8 11 5;9 11 4;10 11 4]; % arrows and weights
    s=1; % the network source
    t=11; % the network sink
    fprintf('The source of the net s=%d\nThe sink of the net t=%d\n',s,t)
    grPlot(V(:,1:2),E,'d','','%d');
    title('\bfThe digraph with weighed edges')
    [dSP,sp]=grShortPath(E,s,t);
    disp('The shortest paths between all vertexes:');
    fprintf('    %2.0f',1:size(dSP,2));
    fprintf('\n');
    for k1=1:size(dSP,1),
      fprintf('%2.0f',k1)
      fprintf('%6.2f',dSP(k1,:))
      fprintf('\n')
    end
    grPlot(V(:,1:2),[sp(1:end-1);sp(2:end)]','d','%d','')
    title(['\bfThe shortest path from vertex ' ...
      num2str(s) ' to vertex ' num2str(t)])
  case 26, % grShortVerPath test
    disp('The grShortVerPath test')
    V=[-8 0; -8 1; -7 0; -7 1; -6 0; -6 1; -6 2; -5 2; -4 1; -3 2;
       -3 1; -2 0; -2 1; -1 2;  0 1;  0 2];
    E=[1 3; 2 4; 3 5; 3 6; 4 6; 4 7; 5 9; 6 9; 7 8; 8 9; 8 10; 9 11;
```

```
       11 12; 11 13; 10 14; 13 14; 13 15; 14 16];
    V(:,3)=[5;9;1;2;2;4;9;9;1;4;6;5;7;7;3;5];
    grPlot(V,E,'d','','',3);
    title('\bfThe digraph with weighted vertices')
    [dMWP,ssp]=grShortVerPath(E,V(:,3)); % solution
    E1=repmat(dMWP(2:end-1),2,1);
    E2=[dMWP(1); E1(:); dMWP(end)];
    E3=reshape(E2,2,length(E2)/2)';
    disp('The Minimal Vertexes Path:')
    fprintf('%d   ',dMWP)
    fprintf('\nThe total weight = %d\n',ssp)
    grPlot(V,E3,'d','','',3)
    title('\bfThe Minimal Vertexes Weight Path')
  case 27, % grTranClos test
    disp('The grTranClos test')
    V=[0 0;1 1;1.2 0.2;1 -1;2 1.2;2.2 0.2;2 -1.2;3 1;...
       3.2 -0.2;3 -1;4 0]; % vertexes coordinates
    E=[1 2;1 3;1 4;2 3;3 4;2 5;...
       2 6;3 6;3 7;4 7;5 6;6 7;...
       5 8;6 8;6 9;7 9;7 10;8 9;...
       9 10;8 11;9 11;10 11]; % arrows
    grPlot(V,E,'d','%d','');
    title('\bfThe initial digraph')
    Ecl=grTranClos(E); % solution
    grPlot(V,Ecl,'d','%d','');
    title('\bfThe transitive closure for the digraph E')
  case 28, % grTravSale test
    disp('The grTravSale test')
    C=[0 3 7 4 6 4;4 0 3 7 8 5;6 9 0 3 2 1;...
       8 6 3 0 9 8;3 7 4 6 0 4;4 5 8 7 2 0];
    disp('The distances between cities:')
    fprintf('  %18.0f',1:size(C,2))
    fprintf('\n')
    for k1=1:size(C,1),
      fprintf('%2.0f',k1)
      fprintf('%20.12f',C(k1,:))
      fprintf('\n')
    end
    [pTS,fmin]=grTravSale(C);
    disp('The order of cities:')
    fprintf('%d   ',pTS)
    fprintf('\nThe minimal way =%3.0f\n',fmin)
  otherwise,
    error('Select the test')
end
```