

# Employing Energy-Aware Multipath Geographical Routing to Bypass Routing Gaps in Wireless Sensor Networks

Miss. Bichkar Poonam Vijay Rukmini<sup>1</sup>, Prof. Ashish Zanjade<sup>2</sup>

<sup>1</sup>PG Scholar, YTIET, University of Mumbai

<sup>2</sup>Assistant Professor, YTIET, University of Mumbai

## ABSTRACT

Geo-Routing also called Geographic Routing or Position-based Routing is a routing principle that relies on geographic position information. With this information, a message can be routed to the destination without knowledge of the network topology or a prior route discovery. Energy conservation and load balance are two important goals in designing routing protocols for Wireless Sensor Networks (WSNs) due to two challenges. First, the sensor nodes are usually powered only by batteries but expected to operate for a long period; Second, it is infeasible and costly to replace or recharge batteries once sensor nodes have been deployed. Currently, existing geographic routing protocols tend to walk along only one side of the routing holes to recover the route. Furthermore, these protocols cannot guarantee that all packets are delivered in an energy-efficient manner once encountering routing holes. In this report, we focus on addressing these issues and propose an Energy-aware Multipath Geo-Routing (EMGR) protocol for better route recovery from routing holes. EMGR adaptively utilizes the location information, residual energy, and the characteristics of energy consumption to make routing decisions, and dynamically exploits two node-disjoint anchor lists, passing through two sides of the routing holes, to shift the routing path for load balance. The proposed EMGR is applicable to resource-constrained WSNs with routing holes.

**KEYWORDS:** Wireless sensor networks, geographic routing, energy-aware routing, anchor list, routing hole.

## 1. INTRODUCTION

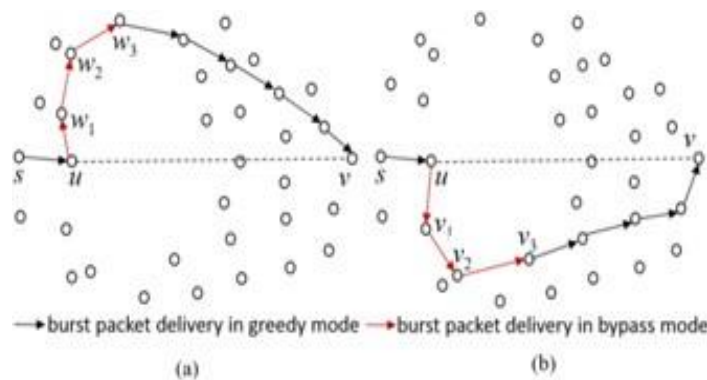
### 1.1 OVERVIEW

ENERGY conservation and load balance are two important goals in designing routing protocols for Wireless Sensor Networks (WSNs) due to two challenges.

First, the sensor nodes are usually powered only by batteries but are expected to operate for a long period. Second, it is infeasible and costly to replace or recharge batteries once sensor nodes have been deployed. Notice that the routing holes, referring to an area free of nodes closer to destination, are hardly avoided in WSNs in various actual geographical environments such as puddles, obstacles, and buildings, and this incurs additional energy expenditure used for data delivery. In this paper, therefore, we focus on designing energy-aware geographic routing protocols regarding how to bypass routing holes for resource-constrained WSNs, which can achieve both energy efficiency by selecting the energy-optimal forwarders and load balance by employing two node-disjoint anchor lists passing through two sides of

the routing holes to shift routing path.

Geographic routing, also referred to as position-based [2] or localized routing [16], has been regarded as an attractive approach for resource-constrained WSNs, since it exploits local location information instead of global topology information for data delivery. It is based on the prerequisite that the nodes know their actual or virtual locations, which can be made available either through a Global Position System (GPS) receiver or through some other ways [2], and exchange such information with neighbors periodically or actively. Being almost stateless and distributed, geographic routing does not require dissemination of route establishment information and maintenance of routing tables at each node, thus making it efficient, scalable and promising for WSNs. A recent detailed performance evaluation and comparison on geographic routing is given. Energy conservation and load balance are two important goals in designing routing protocols for Wireless Sensor Networks (WSNs) due to two challenges [1], [2]. First, the sensor nodes are usually powered only by batteries but expected to operate for a long period; Second, it is infeasible and costly to replace or recharge batteries once sensor nodes have been deployed. Notice that the routing holes, referring to an area free of nodes closer to destination [2], [3], [4], are hardly avoided in WSNs in various actual geographical environments such as puddles, obstacles, and buildings, and this incurs additional energy expenditure used for data delivery. In this paper, therefore, we focus on designing energy-aware geographic routing protocols regarding how to bypass routing holes for resource-constrained WSNs, which can achieve both energy efficiency by selecting the energy-optimal forwarders and load balance by employing two node-disjoint anchor lists passing through two sides of the routing holes to shift routing path.



**Figure 1.1: Face routing bypassing the routing holes indicated by red arrows: (a) employing the right-hand rule, and (b) employing the left-hand rule.**

Fig. 1.1 elaborates an example of face routing by employing such two rules, which detours a routing hole by forwarding the data to the node that is first traversed by the arriving edge of the packet counterclockwise or clockwise, shown in Fig. 1.1(a) and Fig. 1.1(b), respectively. There are at least two paths along the two sides of the routing hole, i.e.,  $u-w_1-w_2-w_3 \rightarrow v$  in Fig. 1(a) and  $u-v_1-v_2-v_3 \rightarrow v$  in Fig. 1(b), provided for route recovery for data delivery. However, the right-hand rule only allows for counterclockwise bypass traversal and the left-hand rule only allows for clockwise bypass traversal, meaning that both of them only walk along one side of the routing holes for route recovery. Beyond face routing over planar networks, there are also other bypass approaches to recover route from routing holes [7]. A comprehensive survey on various bypass approaches is given in [5].

These proposed protocols are simple to implement but have a common shortcoming that they walk along only one side of the routing holes to recover the Such approaches will make the traffic load converged on the boundary of the routing holes, and consequently achieve suboptimal network performance such as

longer delivery delay and lower delivery ratio during routing packets. Furthermore, all of them cannot guarantee that all packets are delivered in an energy-efficient manner [18], [20], [24], [25], since these protocols are more inclined to route data along the boundary of the routing holes or tend to generate long path once encountering the routing holes during routing data, thereby consuming additional energy. In this paper, we propose an energy-aware multipath geographic routing protocol called EMGR for better route recovery from routing holes. The above-mentioned three issues are taken into account in our routing design, thus both energy conservation and load balance can be achieved.

## 1.2 OBJECTIVES

The main objective of this project is:

- 1.2.1** To establish multipath i.e., dual-path routing following two node-disjoint anchor lists which pass through two sides of the routing holes to route data, preventing data from being forwarded along the boundary of the routing holes. In this way, each data packet is routed to the destination along two different paths in greedy mode only instead of bypass mode, if possible, thereby shortening the routing length and balancing load.
- 1.2.2** To find an efficient forwarder in the presence of node failure in the relay area, by introducing a random shift to the location of sub-destination. Such an approach is feasible, reasonable, and energy-efficient without additional communication overhead.
- 1.2.3** To prove that EMGR is anchor list node-disjoint and routing loop-free, and draw out its essential characteristics in terms of time complexity for anchor list building and successful routing probability.
- 1.2.4** We extend EMGR into three-dimensional (3D) sensor networks to provide energy-aware routing for routing hole detour.
- 1.2.5** To evaluate the performance of EMGR and its extension in a variety of communication scenarios, including varied communication sessions, network densities, and routing hole sizes. The results show that EMGR outperforms the existing energy-aware geographic routing protocols.

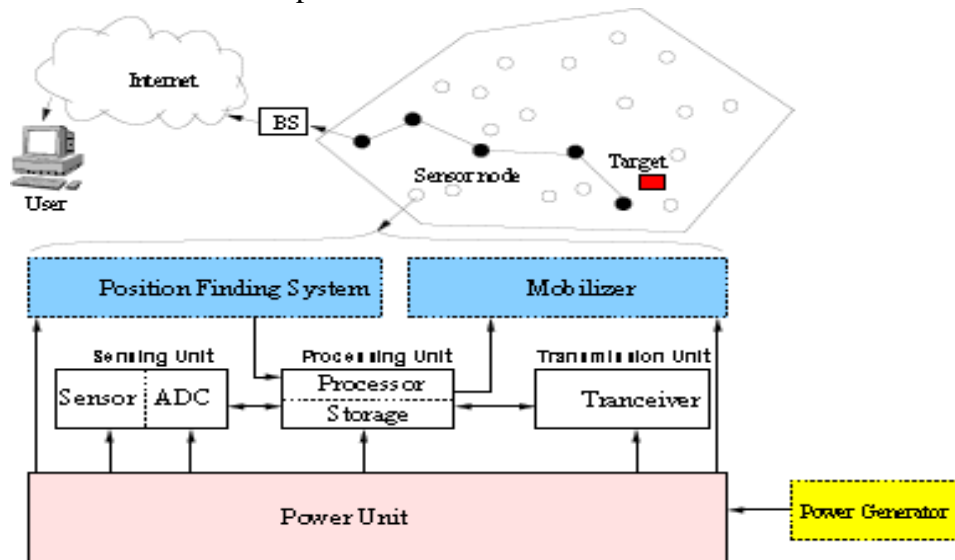
## 1.3 PROJECT SCOPE

The scope of the system is help to analyze the comments as well analyze the sentiments provided by different users from different locations. Here we review the feedback of users which provided by web video while we uploading. here we decide the rating of the web video which means the popularity of video.

## 1.4 WIRELESS SENSOR NETWORK (WSN)

A large number of these disposable sensors can be networked in many applications that require unattended operations. A Wireless Sensor Network (WSN) contains hundreds or thousands of these sensor nodes. These sensors can communicate either among each other or directly to an external base-station (BS). A greater number of sensors allows for sensing over larger geographical regions with greater accuracy. Figure 1.2 shows the schematic diagram of sensor node components. Basically, each sensor node comprises sensing, processing, transmission, mobilizer, position-finding system, and power units (some of these components are optional like the mobilizer). The same figure shows the communication architecture of a WSN. Sensor nodes are usually scattered in a sensor field, which is an

area where the sensor nodes are deployed. Sensor nodes coordinate among themselves to produce highly quality information about the physical environment. Each sensor node bases its decisions on its mission, the information it currently has, and its knowledge of its computing, communication, and energy resources. Each of these scattered sensor nodes has the capability to collect and route data either to other sensors or back to an external base station(s)<sup>1</sup>. A base-station may be a fixed node or a mobile node capable of connecting the sensor network to an existing communications infrastructure or to the Internet where a user can have access to the reported data.



**Figure 1.2: The Component of Sensor Node.**

Networking unattended sensor nodes may have a profound effect on the efficiency of many military and civil applications such as target field imaging, intrusion detection, weather monitoring, security and tactical surveillance, distributed computing, detecting ambient conditions such as temperature, movement, sound, light, or the presence of certain objects, inventory control, and disaster management. Deployment of a sensor network in these applications can be in a random fashion (e.g., dropped from an airplane) or can be planted manually (e.g., fire alarm sensors in a facility). For example, in a disaster management application, a large number of sensors can be dropped from a helicopter. Networking these sensors can assist rescue operations by locating survivors, identifying risky areas, and making the rescue team more aware of the overall situation in the disaster area. In the past few years, intensive research that addresses the potential of collaboration among sensors in data gathering and processing and, in the coordination, and management of the sensing activity were conducted. However, sensor nodes are constrained in energy supply and bandwidth. Thus, innovative techniques that eliminate energy inefficiencies that would shorten the lifetime of the network are highly required. Such constraints combined with a typical deployment of large number of sensor nodes pose many challenges to the design and management of WSNs and necessitate energy-awareness at all layers of the networking protocol stack. For example, at the network layer, it is highly desirable to find methods for energy-efficient route discovery and relaying of data from the sensor nodes to the BS so that the lifetime of the network is maximized.

### 1.5 CHALLENGES

Routing in WSNs is very challenging due to the inherent characteristics that distinguish these

networks from other wireless networks like mobile ad hoc networks or cellular networks.

**1.5.1** Routing around connectivity holes.

**1.5.2** Efficiency in Relay Selection.

**1.5.3** Localization Errors Recovery

**1.5.4** Due to the relatively large number of sensor nodes, it is not possible to build a global addressing scheme for the deployment of a large number of sensor nodes as the overhead of ID maintenance is high. Thus, traditional IP-based protocols may not be applied to WSNs. Sensor nodes that are deployed in an ad hoc manner need to be self-organizing as the ad hoc deployment of these nodes requires the system to form connections and scope with the resultant nodal distribution especially since the operation of the sensor networks is unattended. In WSNs, sometimes getting the data is more important than knowing the IDs of which nodes sent the data.

**1.5.5** In contrast to typical communication networks, almost all applications of sensor networks require the throw of sensed data from multiple sources to a particular BS.

**1.5.6** Sensor nodes are tightly constrained in terms of energy, processing, and storage capacities. Thus, they require careful resource management.

**1.5.7** In most application scenarios, nodes in WSNs are generally stationary after deployment except for, may be, a few mobile nodes. Nodes in other traditional wireless networks are free to move, which results in unpredictable and frequent topological changes. However, in some applications, some sensor nodes may be allowed to move and change their location (although with very low mobility).

**1.5.8** Sensor networks are application specific, i.e., design requirements of a sensor network change with application. For example, the challenging problem of low-latency precision tactical surveillance is different from that required for aperiodic weather-monitoring task.

**1.5.9** Position awareness of sensor nodes is important since data collection is normally based on the location. Currently, it is not feasible to use Global Positioning System (GPS) hardware for this purpose.

**1.5.10** Data collected by many sensors in WSNs is typically based on common phenomena, hence there is a high probability that this data has some redundancy. Such redundancy needs to be exploited by the routing protocols to improve energy and bandwidth utilization.

## **2. EXISTING SYSTEM AND DISADVANTAGES**

### **2.1 GEOGRAPHIC ROUTING PROTOCOLS**

Most geographic routing protocols, such as GFG and GPSR, tend to route data along the boundary of the routing holes for route recovery by employing face routing scheme [4], [6], [12]. Once the bypass mode is involved in the routing, these combinatorial protocols cannot guarantee that all packets are delivered in an energy-efficient manner since they tend to generate long paths and hence consume additional energy during routing packets. Beyond face routing, there are also other approaches [5], [7], [11] used to recover the route. The basic idea of these approaches is to localize routing holes before data delivery and then to derive a detour path with anchor nodes, such that the data is routed to desired destination in an energy-aware manner. However, all of the above-mentioned approaches only walk along one side of the routing holes to recover the route, thus making the load converge on the boundary of the routing holes.

## 2.2 DISADVANTAGES

The main Disadvantages of Existing System are:

- A large number of energy-aware geographic routing protocols
- A limited number of energy-aware geographic routing protocols have taken into consideration the load balance for route recovery from routing holes.
- Several power-aware techniques that attempt to balance the traffic among the nodes to extend the network lifetime.
- Balancing the traffic to conserve energy, by randomly shifting the location of each anchor node.
- Sensor nodes are energy-constrained and battery recharging is often infeasible.
- Time Consuming.

## 3. LITERATURE SURVEY

In the past decade, a variety of geographic routing protocols have been proposed, such as [3], [4], [6], [7]. Below several researches on them are shown.

1. **Reference [2] [5]** to address the routing hole issues occurring in WSNs. A comprehensive survey of geographic routing protocols regarding how to bypass the routing holes can be found in [2], [5]. Commonly, these proposed protocols utilize greedy mode to route data packets as far as possible and switch to bypass mode for route recovery once encountering a routing hole.
2. **Reference [9]** Currently, most geographic routing protocols, such as Greedy Perimeter Stateless Routing (GPSR) [4], Greedy Face-Greedy (GFG) [6], and Greedy Other Adaptive Face Routing (GOAFR) [9], tend to exploit face routing scheme to bypass routing holes, while doing not detect them before data delivery.
3. **Reference [10]** These approaches [10], [11], will make the traffic load converged on the boundary of the routing holes, and consequently achieve suboptimal network performance such as longer delivery delay and lower delivery ratio during routing packets.
4. **Reference [8]** All these approaches guarantee that all packets are delivered in an energy efficient manner, since these protocols are more inclined to route data along the boundary of the routing holes or tend to generate long path once encountering the routing holes during routing data, thereby consuming additional energy.
5. **Reference [12]** Specifically, Wu et al. [12] proposed several power-aware techniques that attempt to balance the traffic among the nodes to extend the network lifetime.
6. **Reference [13]** In [13], each forwarder first establishes a sub-destination among its one-hop neighbors, and then routes the data to it through an intermediary node or alters the sub destination if this will preserve power.
7. **Reference [14]** Zhao et al. [14] suggested balancing the traffic to conserve energy, by randomly shifting the location of each anchor node. The work in [28] proposed an energy-aware multipath geographic routing to balance load, by exploiting a dynamic anchor list passing through one side of the routing holes to shift routing path.
8. **Routing Algorithms Based on the Cluster:** The routing algorithms based on the cluster include LEACH-L [31], the Clustering Vector-Based Forwarding (CVBF) algorithm, and a popular clustering algorithm for time series data transformed by a multi-resolution dimensionality reduction method called as I-K<sub>means</sub>. The drawback of those algorithms based on the cluster is that the energy balance is hard to solve. In this paper, He proposes a shortest path routing protocol based on the

vertical angle (SPRVA), which exhibits short delay and high energy efficiency. SPRVA is based on DBR. In the network structure, many sink nodes are uniformly distributed on the surface. Because UANs use acoustic communication, the end-to-end delay is mainly composed of propagation delay. To reduce propagation delay, SPRVA first selects the next-hop node based on the angle between the propagation direction and the vertical direction. A smaller angle leads to a shorter propagation distance, which results in less propagation loss. The main priority is calculated by combining the residual energy and the angle.

#### 4. PROBLEM DEFINITION

Generally, geographic routing utilizes greedy mode to route data packets when it can find a neighbor closer to the destination than the current forwarder and switches to bypass mode once the data packets encounter a routing hole, where there is no such node closer to the destination than the current forwarder. To achieve our design goal, there are at least three issues to be addressed for the current geographic routing. How to detect routing holes before data delivery, since the routing holes occurring in routing will generate long paths and hence consume additional resources. How to bypass routing holes for load balance. How to select the energy-aware forwarders and therefore guarantee that all packets are delivered in an energy-efficient manner for resource-constrained WSNs.

1. How to detect routing holes before data delivery, since the routing holes occurring in routing will generate long paths and hence consume additional resources.
2. How to bypass routing holes for load balance.
3. How to select the energy-aware forwarders and therefore guarantee that all packets are delivered in an energy-efficient manner for resource-constrained WSNs.

#### 5. PROPOSED SYSTEM

The proposed EMGR protocol for bypassing routing holes in WSNs. First, the EMGR architecture is presented, and then how to obtain an anchor list is introduced. Finally, we formulate how to deliver messages in an energy-efficient manner.

##### 5.1 ENERGY-AWARE MULTIPATH GEOGRAPHIC ROUTING

The following are the function entity that plays a very important role in this project which is shown in the system architecture

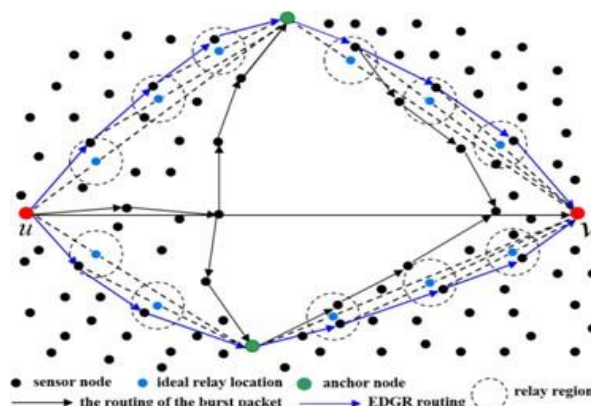


Figure 5.1: EMGR architecture

The main mechanism of EMGR is to employ multiple esp. two-node disjoint anchor lists to guide packet delivery and select the nodes with more residual energy from energy-aware relay regions as forwarders

for energy conservation. Thus, the data packets are likely routed to the anchor nodes and their destinations along two paths at an energy-efficient cost. Figure 5.1 above illustrates the network architecture of EMGR. The operation of EMGR is mainly divided into two phases: anchor list obtaining and data dissemination. In the first phase, the proposed EMGR scheme uses an adaptive approach to obtain two anchor lists based on the projected distance of nodes being involved in bypass mode. In the latter phase, the proposed EMGR utilizes geographic-information, the residual energy and the characteristics of energy consumption to make routing decisions, and then unicasts the packet to the established next-hop forwarder. If there is no node in the relay region, the current forwarder then introduces a random shift to the location of the sub-destination to continue data delivery.

There are three kinds of packets: beacon packet, burst packet, and data packet in our scheme. The beacon packet is used to exchange location information and residual energy among neighbors, while burst packet is used for finding the anchor lists. The above figure presents the format of the burst packet. Specifically, it includes anchor lists, the locations of source node and destination. The anchor list contains a series of anchor nodes, and a flag field which indicates whether this packet bypasses the routing holes by employing the right-hand rule or left-hand rule. In addition, it includes a temporary void node in each bypass mode but is deleted finally if it violates the determined rules of anchor nodes. Here, the **void node** is defined as the node that switches to bypass mode from greedy mode, i.e., the node (e.g., node  $u$  in Fig. 5.1) that cannot find a neighbor closer to the destination than itself, even though there is a path from the source node to destination in the network.

## 5.2 ANCHOR LIST OBTAINING

Given source node  $u$ , it starts preparing for data dissemination by building two anchor lists. First, it adaptively broadcasts a beacon packet to its neighbors at the maximum transmission power for announcing its location information and residual energy. Once a neighbor receives this packet, then stores such information and broadcasts a new beacon packet to its neighbors at the maximum transmission power. This process will repeat periodically among all nodes in the network, such that each node can obtain the location information and residual energy of its neighbors within their maximum transmission range. Once receiving location information and residual energy of all neighbors, source node  $u$  initiates and sends a burst packet to destination  $v$ . For any forwarder  $w$ , it uses greedy mode to forward this burst packet whenever possible and switches to bypass mode when encountering a routing hole. The bypass mode begins from the void node. For any void node  $w$  in the  $j$ th bypass mode, it first adds itself into the burst packet header (i.e., anchor list) for its downstream forwarders to decide whether to return to greedy mode. Then, it takes into consideration the following two cases to make routing decisions on how to bypass this routing hole.

**Case I.** If the flag is void, it copies this burst packet and then simultaneously uses the right-hand rule and left-hand rule to bypass this hole, and sets the flag of two burst packets  $r$  and  $l$ , respectively, for their continued delivery relayed by the subsequent nodes in accordance with it.

**Case II.** If the flag is not void, this node and its subsequent relay nodes use the right-hand rule indicated by  $r$  or the left-hand rule indicated by  $l$  to detour this routing hole.



### 5.3 DATA DISSEMINATION

In our scheme, source node  $u$  and each forwarder regard anchor nodes as sub-destinations. Before data delivery, node  $u$  randomly selects an anchor list and then embeds the anchor nodes into the data packet header.

### 5.4 SYSTEM ANALYSIS

#### 5.4.1 Node-disjoint Anchor List

In EMGR, two anchor lists built by employing right-hand rule and left-hand rule are node-disjoint if such two anchor lists can be found.

#### 5.4.2 Time Complexity

The time complexity for EMGR to build two anchor lists is  $\theta(n)$ . The approaches, most similar to EMGR to build anchor lists, mainly include Projection Distance-based Anchor (PDA+) and Energy-aware Multipath Geographic Routing (EMGR). However, both of them all only build an anchor list, with the time complexity  $\theta(n_1)$  (or  $\theta(n_2)$ ), and  $\theta(n_1)$  (or  $\theta(n_2)$ ), respectively, to recover the route from routing holes. The results reveal that, similar to PDA+ and EMGR, EMGR achieves the linear time complexity to build anchor lists.

#### 5.4.3 Guaranteed Delivery

EMGR is routing loop-free. EMGR combines bypass mode and greedy mode to forward the data packets, by building the anchor list to avoid the routing holes. In this way, EMGR prevents data packet from being forwarded along the boundary of holes, thus each data packet is routed to destination node only in greedy mode instead of bypass mode if possible, as shown in fig 5.2

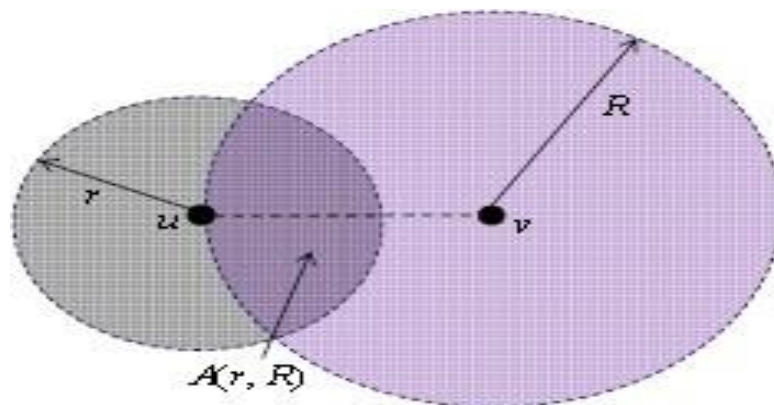


Figure 5.2: The advance region for the current forwarder  $u$  to destination  $v$ .

## 6. METHODOLOGY

### 6.1 TECHNOLOGY USED

It is considered that no two nodes are located at the same location, as in and so on. All sensor nodes are distributed in the network following Poisson distribution. Each sensor node knows its own location through an internal GPS device or a separate calibration process, and knows the location of neighbors and their residual energy within its maximum transmission power by exchanging beacon messages. The source node can obtain the location information of packet destinations by some destination location services. The location of a node acts as its ID and its network address. Therefore, there is no need for a separate ID establishment protocol. We only consider bidirectional links, and assume that each sensor node can adjust its transmission power from 0 to its maximum transmission power.

EMGR, for instance, EDGR (Energy-aware Dual-path Geo-Routing) establishes dual-path routing following two node-disjoint anchor lists that pass through two sides of the routing holes to route data, preventing data from being forwarded along the boundary of the routing holes. In this way, each data packet is routed to the destination along two different paths in greedy mode only instead of bypass mode, if possible, thereby shortening the routing length and balancing load.

EMGR proposes a novel alternative approach to find efficient forwarder in the presence of node failure in the relay area, by introducing a random shift to the location of sub destination. Such an approach is feasible, reasonable, and energy-efficient without additional communication overhead.

We prove that EMGR is anchor list node-disjoint and routing loop-free, and draw out its essential characteristics in terms of time complexity for anchor list building and successful routing probability.

We extend EMGR into three-dimensional (3D) sensor networks to provide energy-aware routing for routing hole detour.

## 6.2 BUILDING ANCHOR LIST

### 6.2.1 Algorithm

In order to avoid route failure in data delivery between source node and its sub destination, the last sub-destination and destination, and two adjacent sub-destinations, node  $u$  will send a new burst packet to the established sub-destinations and then to destination  $v$ , according to the right-hand rule or the left-hand rule indicated by the flag in the anchor list, to check whether a routing hole exists. If this is true, a new anchor node determined. The action for routing hole acknowledges from source node  $u$  to destination  $v$  going through the established sub destinations will be done even more than once, depending on the network density. By employing right-hand rule and left-hand rule simultaneously to walk along two sides of the all-routing holes, our proposed approach can obtain two node disjoint anchor lists to guide packet delivery. With such two anchor lists, the data packets are likely routed to destinations along two different paths, thus achieving load balance.

*The pseudo-code of building the anchor list is shown in Algorithm-1.*

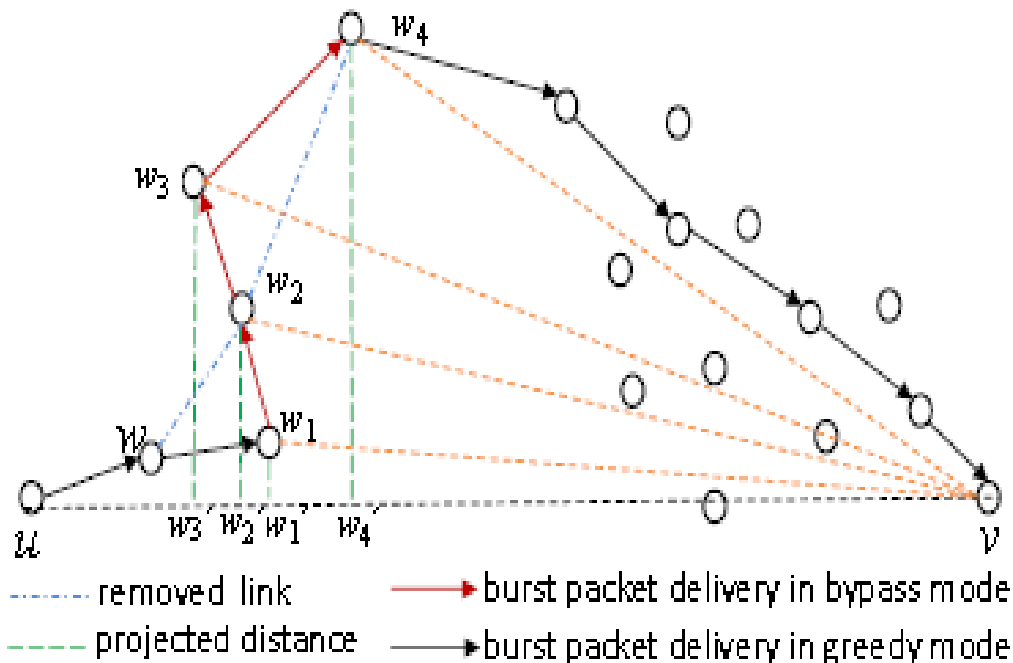
This algorithm is initialized with the required location information and residual energy of neighbors. Then the algorithm sends a burst packet to destination  $v$  and employs both right-hand rule and left-hand rule,  $r$  and  $l$ , to bypass the all-routing holes simultaneously shown in Lines 2-6 and Lines 18-30, and will update anchor list, as illustrated in Lines 7-30, if a routing hole exists between source node and its sub destination, or the last sub destination and its destination, or two adjacent sub destinations. In bypass mode, each candidate anchor node is determined. Finally, two anchor lists are obtained and fed back to source node  $u$  from destination  $v$ .

**Algorithm 1 : Building Anchor List**

```

Require: source node  $u$ , destination  $v$ 
1:  $List(u, v) = [flag, \phi]$ 
Ensure:
2:  $v$  initializes beacon packet exchange
3:  $u$  sends a burst packet to  $v$  attached with  $List(u, v)$ 
4: if  $\forall w_i$  in the bypass mode receives the packet the
5:   UPDATELIST( $w_i, List(u, v)$ )
6: end if
7: if  $u$  receives a feedback packet from  $v$  then
8:   update  $list(u, v) = [flag, a_1 \dots a_j \dots, a_m]$ 
9:   send a burst packet to  $v$  Via  $a_1, a_2, \dots, a_m$ 
10: end if
11: if  $\forall w_i == a_m \in \{a_1, \dots, a_m\}$  receives the packet th
12:   send this packet to  $a_{i+1}$ 
13: else
14:   if  $\forall w_i$  is in the bypass mode then
15:     UPDATELIST( $w_i, List(u, v)$ )
16:   end if
17: end if
18: function UPDATELIST( $w_i, List(u, v)$ )
19:   if  $w_i \in List(u, v)$  then
20:     delete all nodes after  $w_i$  in  $List(u, v)$ 
21:   end if
22:   if  $w_i == w_{f_j}$  or  $w_i == w_{l_j}$  or  $w_{i-1}, w_{i+1}$  are
   bypass mode then
23:     calculate candidate  $w_i$  according to Eq.(2)
24:     add  $w_i$  to  $List(u, v)$ 
25:     send  $List(u, v)$  to  $w_{i+1}$  following  $flag$ 
26:   end if
27:   if  $w_i == v$  then
28:     send  $List(u, v)$  to  $u$ 
29:   end if
30: end function

```



**Figure 6.1: Working Model**

Fig. 6.1 illustrates how to build an anchor list by employing right-hand rule, which is similar to how another anchor list can be built if exists by employing left-hand rule. Therefore, we have only

emphasized it and not depicted the nodes located below the line from source node  $u$  to destination  $v$ . The network is a partial enlarged version of the network in Fig. 6.1, with only one routing hole. The blue dash lines represent removed links to avoid the crossing edges existing according to the right-hand rule, nodes  $w_{01}$ ,  $w_{02}$ ,  $w_{03}$  and  $w_{04}$  are the projected nodes of nodes  $w_1$ ,  $w_2$ ,  $w_3$  and  $w_4$  on the line  $uv$  from source node  $u$  to destination  $v$ . Among all the forwarders, node  $w_1$  is the first node, i.e., void node, which enters bypass mode to forward the burst packet due to no neighbor being closer to destination  $v$ , and node  $w_4$  is the last node involved in such a mode to forward this packet since it can find a neighbor closer to destination  $v$  than node  $w_1$ . There are four nodes, i.e., nodes  $w_1$ ,  $w_2$ ,  $w_3$  and  $w_4$ , involved in bypass mode to build an anchor list, while the other intermediate nodes only forward it. In bypass mode, nodes  $w_1$ ,  $w_2$ ,  $w_3$  and  $w_4$  first calculate their projected nodes on the line  $uv$  and projected distances, indicated by green dash lines in Fig.6.1 and then calculate the candidate anchors according to given Equation Finally, node  $w_4$  is selected as the anchor node. In this way, an anchor list is obtained.

### 6.3 MODULES

The Following Modules are used for implementation:

#### 6.3.1 Form Network

There are three kinds of packets: beacon packet, burst packet, and data packet in our scheme. The beacon packet is used to exchange location information and residual energy among neighbors, while the burst packet is used for finding the anchor lists. Fig.6.1 presents the format of the burst packet. Specifically, it includes anchor lists, the locations of source node and destination. The anchor list contains a series of anchor nodes, and a flag field which indicates whether this packet bypasses the routing holes by employing the right-hand rule or lefthand rule. In addition, it includes a temporary void node in each bypass mode but is deleted finally if it violates the determined rules of anchor nodes. Here, the void node is defined as the node that switches to bypass mode from greedy mode, i.e., the node (e.g., node  $u$  in Fig. 6.1) that cannot find a neighbour being closer to the destination than itself, even though there is a path from the source node to destination in the network.

#### 6.3.2 Anchor List

Given source node  $u$ , it starts preparing for data dissemination by building two anchor lists. First, it adaptively broadcasts a beacon packet to its neighbors at the maximum transmission power for announcing its location information and residual energy. Once a neighbor receives this packet, then stores such information and broadcasts a new beacon packet to its neighbors at the maximum transmission power. This process will repeat periodically among all nodes in the network, such that each node can obtain the location information and residual energy of its neighbors within their maximum transmission range. Once receiving location information and residual energy of all neighbors, source node  $u$  initiates and sends a burst packet to destination  $v$ . For any forwarder  $w$ , it uses greedy mode to forward this burst packet whenever possible and switches to bypass mode when encountering a routing hole. The bypass mode begins from the void node. For any void node  $w$  in the  $j$ th bypass mode, it first adds itself into the burst packet header (i.e., anchor list) for its downstream forwarders to decide whether to return to greedy mode.

### 6.3.3 Data Dissemination

In our scheme, source node  $u$  and each forwarder regard anchor nodes as sub-destinations. Before data delivery, node  $u$  randomly selects an anchor list and then embeds the anchor nodes into the data packet header.

### 6.3.4 Performance

We evaluate the performance of our proposed approach EMGR and its extension via simulation experiments. We first describe our simulation environments and performance metrics, and then evaluate the performance results. Finally, we show the analysis among our proposed approach

## 6.4 GEOGRAPHICAL AND ENERGY-AWARE ROUTING (GEAR)

The protocol, called Geographic and Energy Aware Routing (GEAR), uses energy-aware and geographically-informed neighbour selection heuristics to route a packet towards the destination region. The key idea is to restrict the number of interests in directed diffusion by only considering a certain region rather than sending the interests to the whole network. By doing this, GEAR can conserve more energy than directed diffusion. The basic concept comprises of two main parts- a) Route packets towards a target region through geographical and Energy-aware neighbour selection. b) Disseminate the packet within the region.

## 6.5 PARAMETERS OF QOS NETWORKS

Different services require different QoS parameters Multimedia Bandwidth, delay jitter & delay, Emergency services., Network availability, Group communications, Battery life.

Generally, the important parameters are:

Bandwidth, Delay jitter, Battery charge, Processing power, Buffer space, delay jitter: packet delay variation.

## 7. SYSTEM DESIGN

### 7.1 ACTIVITY DIAGRAM

An activity diagram is a behavioral diagram i.e. it depicts the behavior of a system. An activity diagram portrays the control flow from a start point to a finish point showing the various decision paths that exist while the activity is being executed.

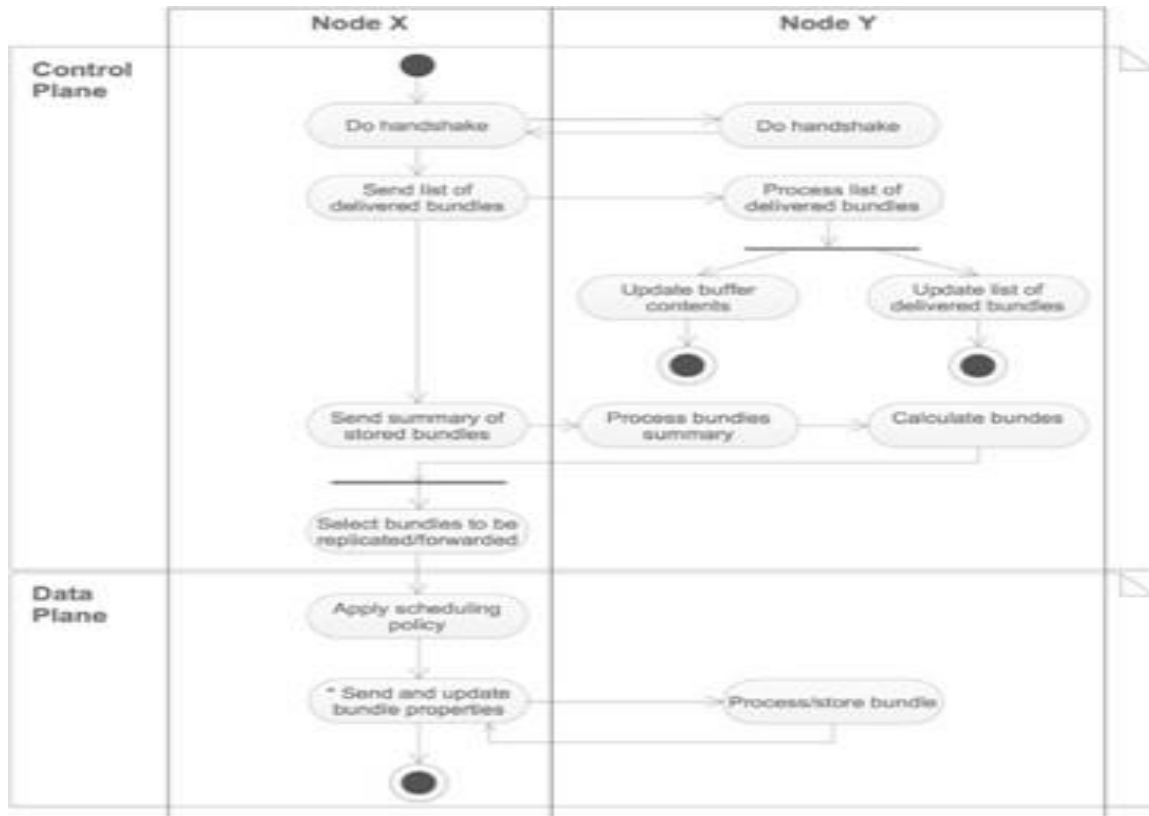


Figure 7.1: Activity Diagram

### 7.2 USE CASE

In this Design use case is a list of actions or event steps typically defining the interactions between a role (known in the Unified Modeling Language (UML) as an actor) and a system to achieve a goal. The actor can be a human or another external system.

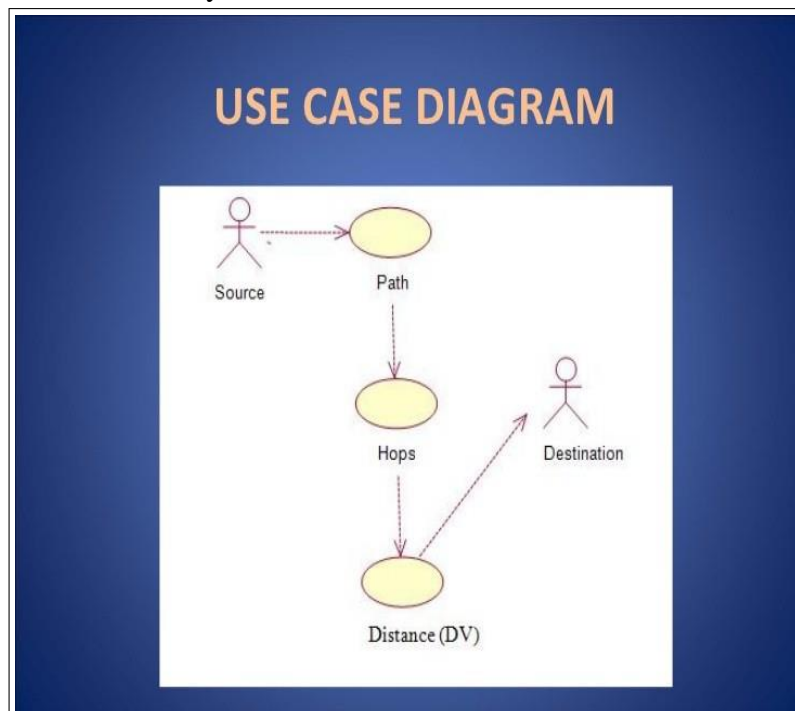


Figure 7.2: Use Case of System

### 7.3 SEQUENCE DIAGRAM

A sequence diagram simply depicts interaction between objects in a sequential order i.e. the order in which these interactions take place. We can also use the terms event diagrams or event scenarios to refer to a sequence diagram. Sequence diagrams describe how and in what order the objects in a system function.

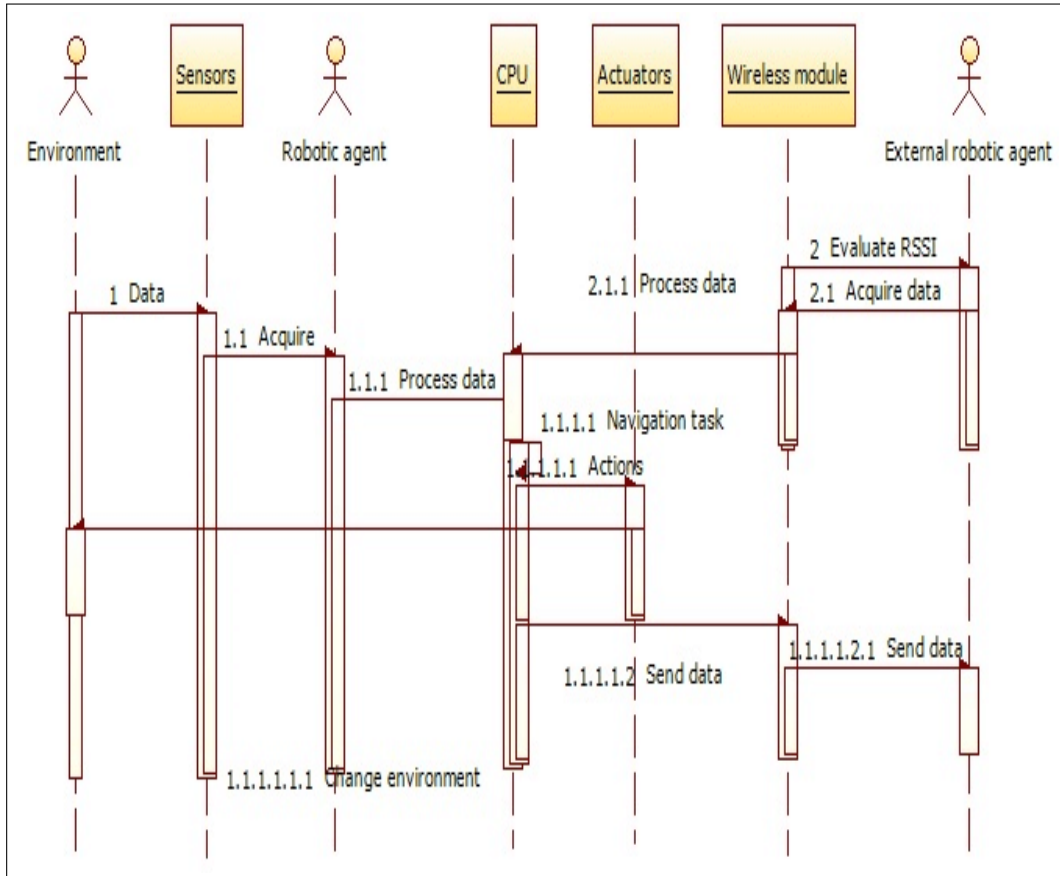


Figure 7.3: Sequence Diagram for system

## 8. SOFTWARE REQUIREMENT SPECIFICATION

### 8.1 REQUIREMENT ANALYSIS

Requirement analysis is for the transformation of operational needs into software description, software performance parameter, and software configuration through use of standard, iterative process of analysis and trade-off studies for understanding what the customer wants analyzing needs, assessing feasibility, negotiating a reasonable solution validating the specification and managing the requirements.

### 8.2 PROJECT SCOPE

1. To instantiate two different data services, and extensively evaluate their performances on two real-world data sets.
2. Under a specific data service, this system provides privacy preservation and verifiability.

### 8.3 OPERATING ENVIRONMENT

#### 8.3.1 Software Requirement

Platform:

1. Operating System: Windows 7 or more
2. IDE: Netbeans 8.2
3. Programming Language: Java (jdk1.7.0 or more)

### 8.3.2 Hardware Requirement

Table 8.1: Hardware Requirements

Sr. No.	Parameter	Minimum Requirement	Justification
1	Processor	Intel Core i3/i5	Available
2	Processor speed	1.9GHz	Available
3	Motherboard	Intel	Available
4	Primary memory	4GB	Available
5	Secondary memory	500GB	Available
6	Monitor	Generic pnp monitor	Available
7	Keyboard	Standard PS/2Keyboard 104 key	Available

#### JDK 1.8 Installation

1. Double click jdk-8-ea-bin-b32-windows-i586 to run the installation program. JDK License dialog displayed. Accept the license in order to install JDK.
2. The JRE Custom setup dialog enables you to choose a custom directory for JRE Files.
3. The complete dialog indicates a successful installation.

#### Net Beans IDE 7.3.1 Installation

To install the software:

1. After the download completes, run the installer. For Windows, the installer executable file has the .exe extension. Double-click the installer file to run it.
2. If you downloaded the All bundle, you can customize your installation. Perform the following steps at the Welcome page of the installation wizard:
3. Click Customize.
4. In the Customize Installation dialog box, make your selections.
5. At the Welcome page of the installation wizard, click Next. At the License agreement page, review the license agreement, click the acceptance check box, and click Next. At the JUnit License Agreement page, decide if you want to install JUnit and click the appropriate option, click Next. At the NetBeans IDE installation page, do the following:
6. Accept the default installation directory for the NetBeans IDE or specify another directory. Note: The installation directory must be empty and the user profile you are using to run the installer must have read/ write permissions for this directory.
7. Accept the default JDK installation to use with the NetBeans IDE or select a different installation from the drop-down list. If the installation wizard did not find a compatible JDK installation to use with the Net-Beans IDE, your JDK is not installed in the default location. In this case, specify the path to an installed JDK and click Next, or cancel the current installation. After installing the required JDK version you can restart the installation.
8. If you are installing Apache Tomcat, on its installation page, accept the default installation directory or specify another installation location. Click Next.



9. At the Summary page, verify that the list of components to be installed is correct and that you have adequate space on your system for the installation.
10. Click Install to begin the installation.
11. At the Setup Complete page, provide anonymous usage data if desired, and click Finish.

## 8.4 FUNCTIONAL REQUIREMENT

The functional requirements of the system are:

1. Fast and efficient system
2. User friendly GUI
3. Reusability
4. Performance
5. System Validation input
6. Proper output

### 8.4.1 Overview of Responsibilities of Developer

- Perform project design and development activities according to customer specifications.
- Work with Manager in developing project plan, budget and schedule.
- Coordinate with management in preparing project proposals and contractual documents.
- Track project progress regularly and develop status reports to management.
- Ensure that project is completed within allotted budget and time lines.
- Follow company policies and safety regulations for operational efficiency.
- Research and recommend new technologies to carry out project developmental tasks.
- Provide assistance to other Developers, perform peer reviews and provide feedback for improvements.
- Develop cost reduction initiatives while maintaining quality and productivity

## 8.5 NON-FUNCTIONAL REQUIREMENT

### 8.5.1 Performance Requirements

- System can produce results faster on 4GB of RAM.
- It may take more time for peak loads at main node.
- The system will be available 100% of the time. Once there is a fatal error, the system will provide understandable feedback to the user.

### 8.5.2 Safety Requirements

- The system is designed in modules where errors can be detected and fixed easily.

### 8.5.3 Software Quality Attributes

- **Reliability:** The Client machine will change the status of data indicating successful data transmission.
- **Usability:** The application should be easy to use through an interactive interface.
- **Maintainability:** The system will be developed using the standard software development conventions to help in easy review and redesigning of the system.
- **Supportability:** The system will be able to support to transfer different types of SQL queries.

- **Portability:** This software is portable to any system with the requirements specified. There must also be a server where the database can be set-up.

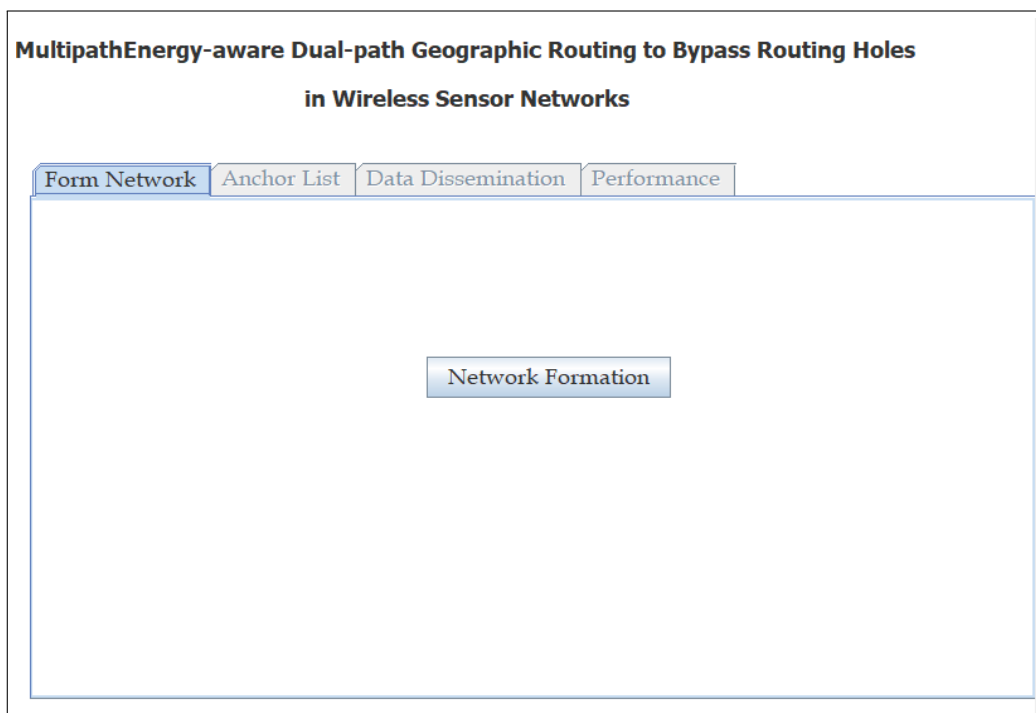
## 8.6 SIMULATION SETUP

The popular network simulation platform, JAVA, is used to evaluate the performance of EMGR. We select anchor list algorithm as the MAC protocol. Unless specially noted, the number of sensor nodes randomly distributed in a 2D area of 1000m.

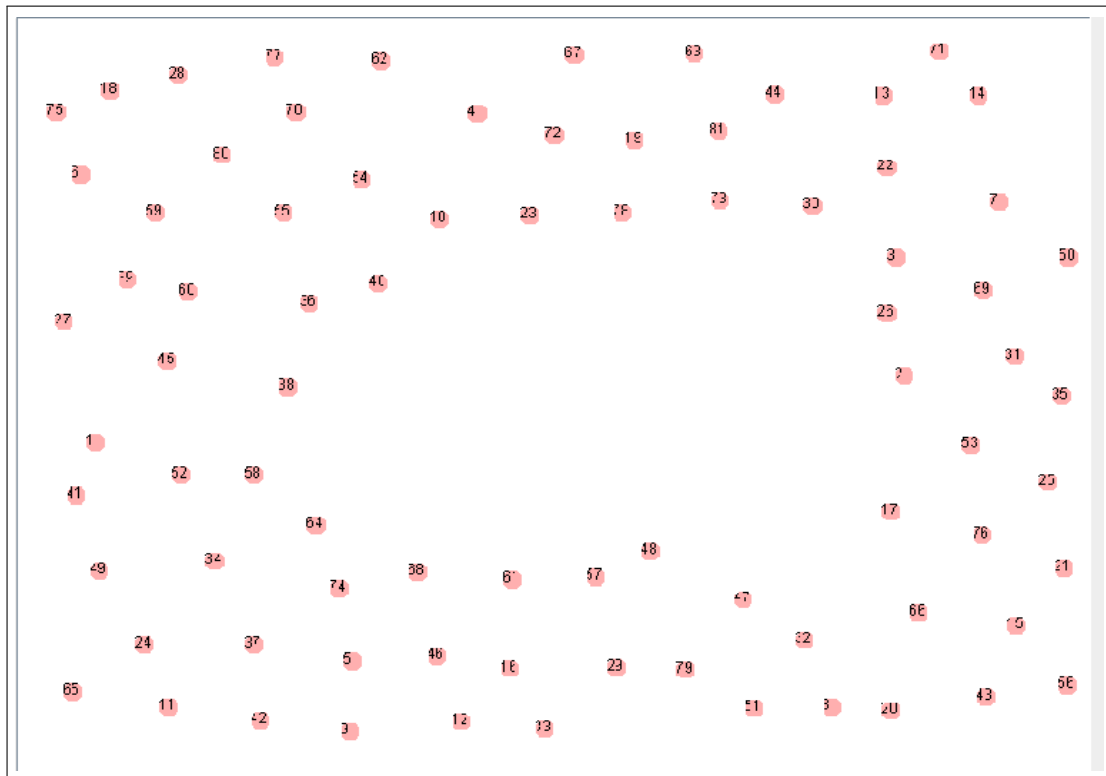
1000m is set to 100, and varies from 200 to 600 only as the network density increases. Each source node randomly generates TCP flows at a varied speed from the range [0:5Mbps; 1Mbps] with the different packet sizes, commonly set as 128 bytes, 256 bytes, or 512 bytes [2], [38]. Each sensor node has an initial energy of 1J, and source-destination pairs are randomly selected in the network. The average performance results are collected from 40 simulation runs, which can make bypass mode more likely to happen in our simulation scenarios. One or two routing holes are set in the center of the network. In each simulation scenario, the size of the routing holes can be varied to ensure that many more communication sessions can pass through them as far as possible. The communication sessions without bypass mode are not considered as our efficient results. Three basic simulation scenarios are designed to evaluate the performance of EMGR.

## 9. IMPLEMENTATION AND RESULTS

### 9.1 HOME



The above screenshot of the project shows the main screen of this project it consist of Four model form network, Anchor list, Data Dissemination and Performance. when we click on form Network that display the Network Formation which the output was shown now.



In the above screenshot we have created the nodes in different location and in between we have shown the different holes. the blank space shows the holes in between the network.

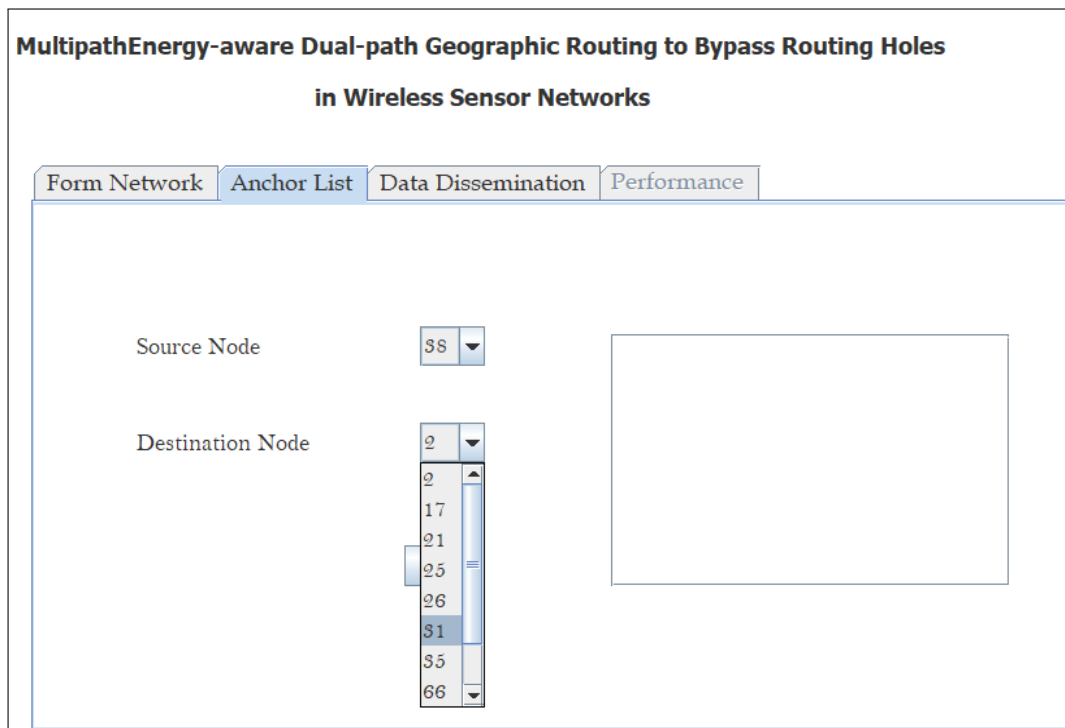
**MultipathEnergy-aware Dual-path Geographic Routing to Bypass Routing Holes  
in Wireless Sensor Networks**

Form Network | **Anchor List** | Data Dissemination | Performance

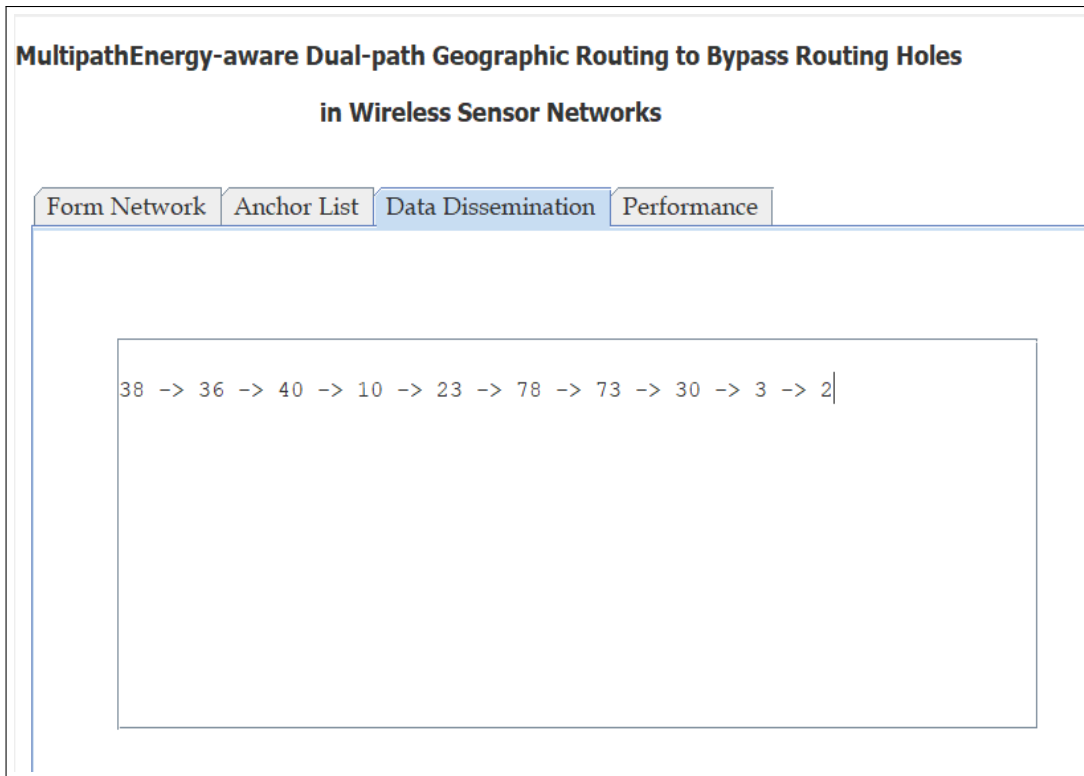
Source Node: 38

Destination Node: 2

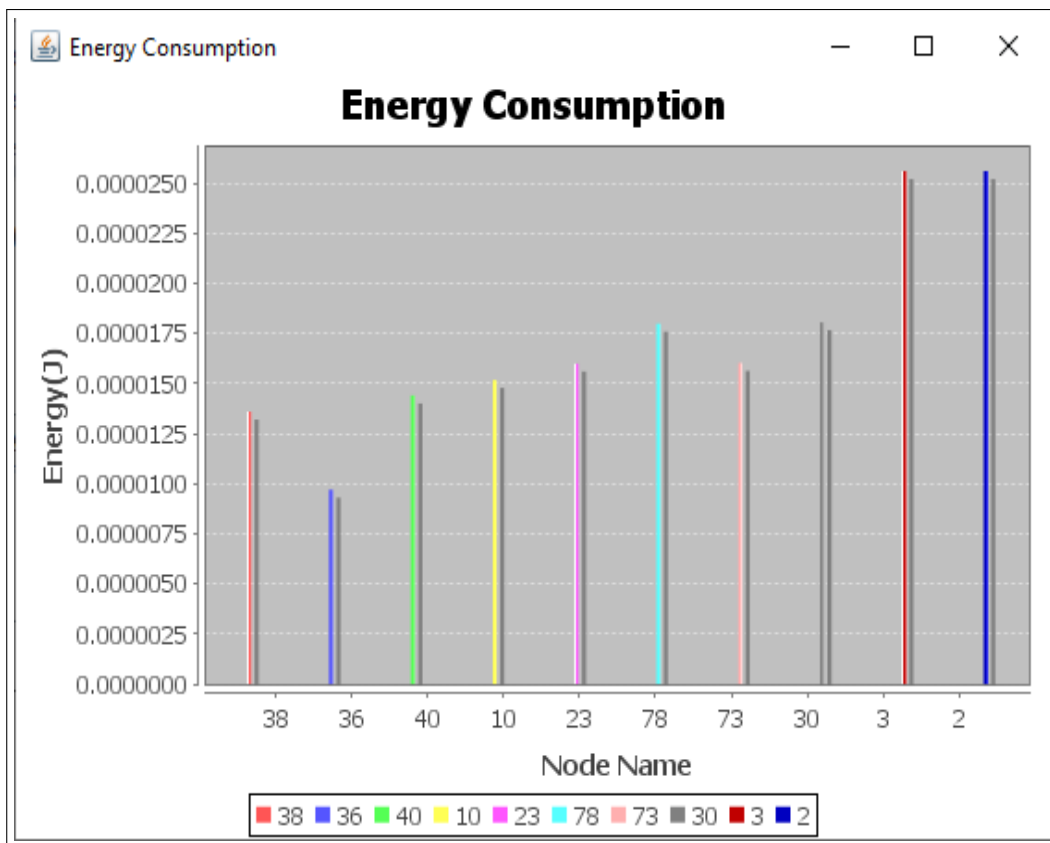
2 | 17 | 21 | 25 | 26 | 31 | 35 | 66



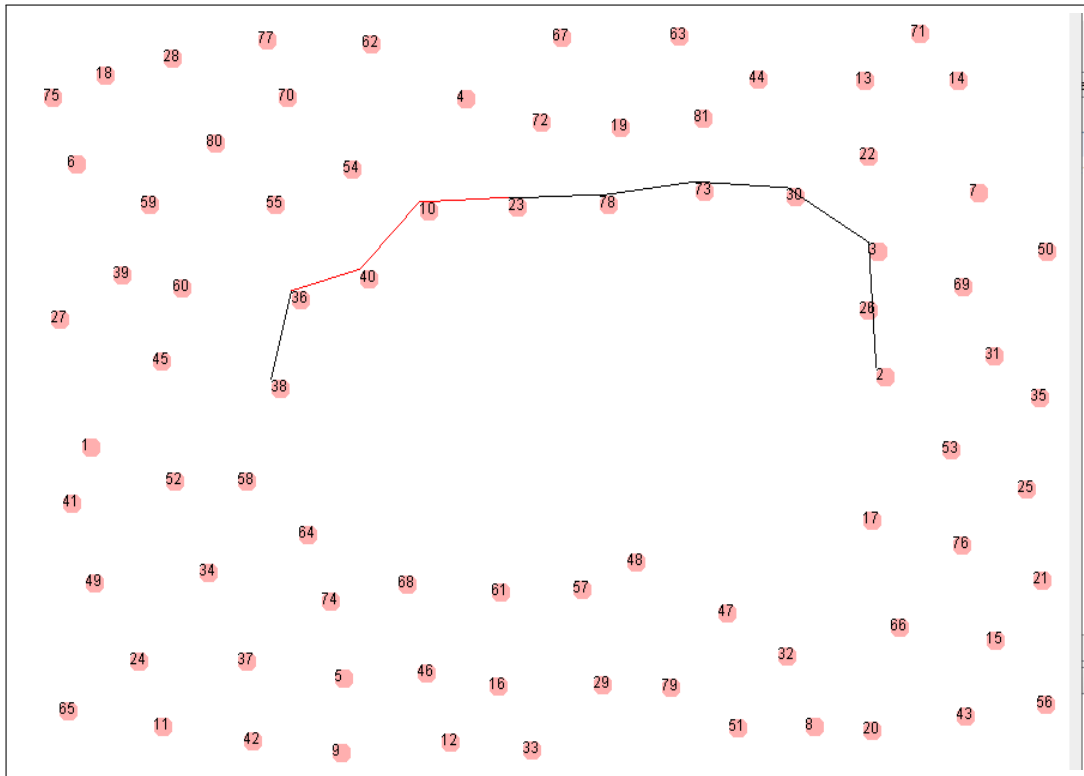
after formation of nodes the next step is to select the source node and destination node. in the drop down we have created some source node and some destination node.



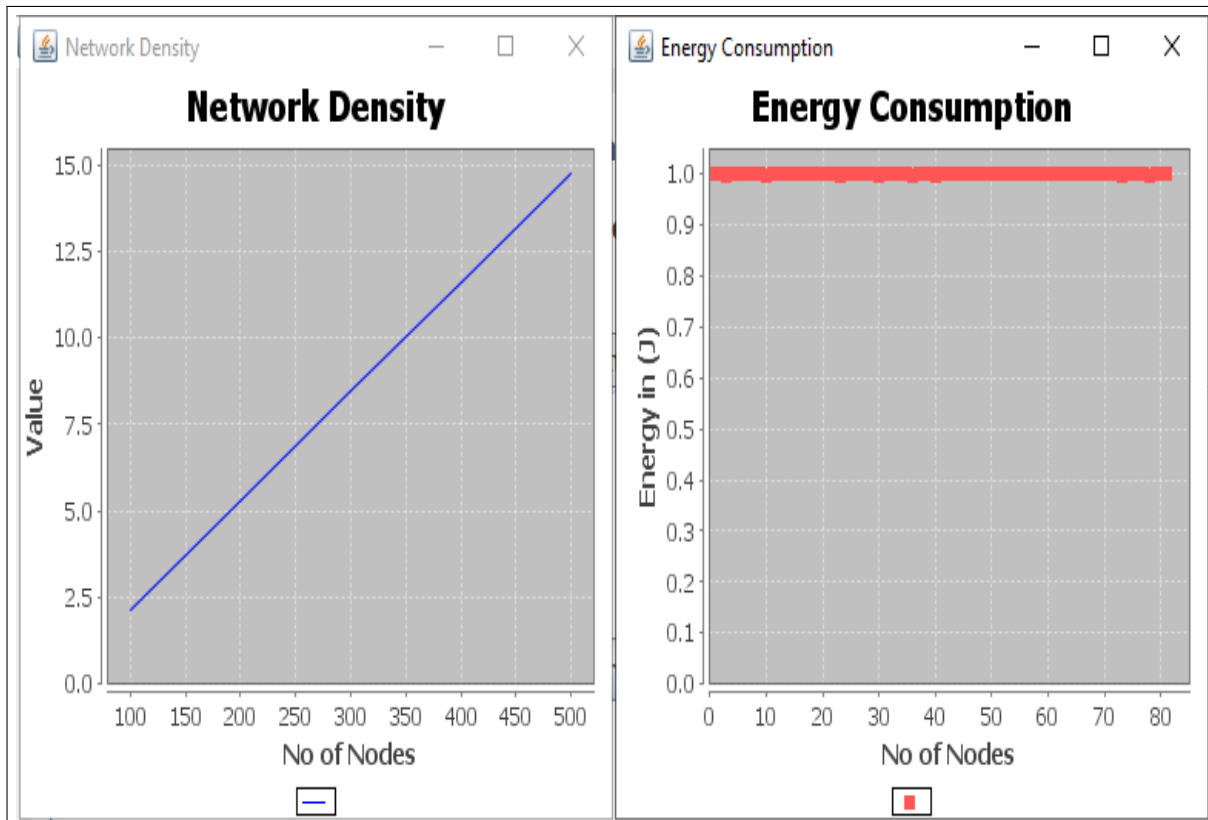
after selecting the source and destination node the system has calculate the pathusing the algorithm and bypassing the holes in routing and shows the correct path.



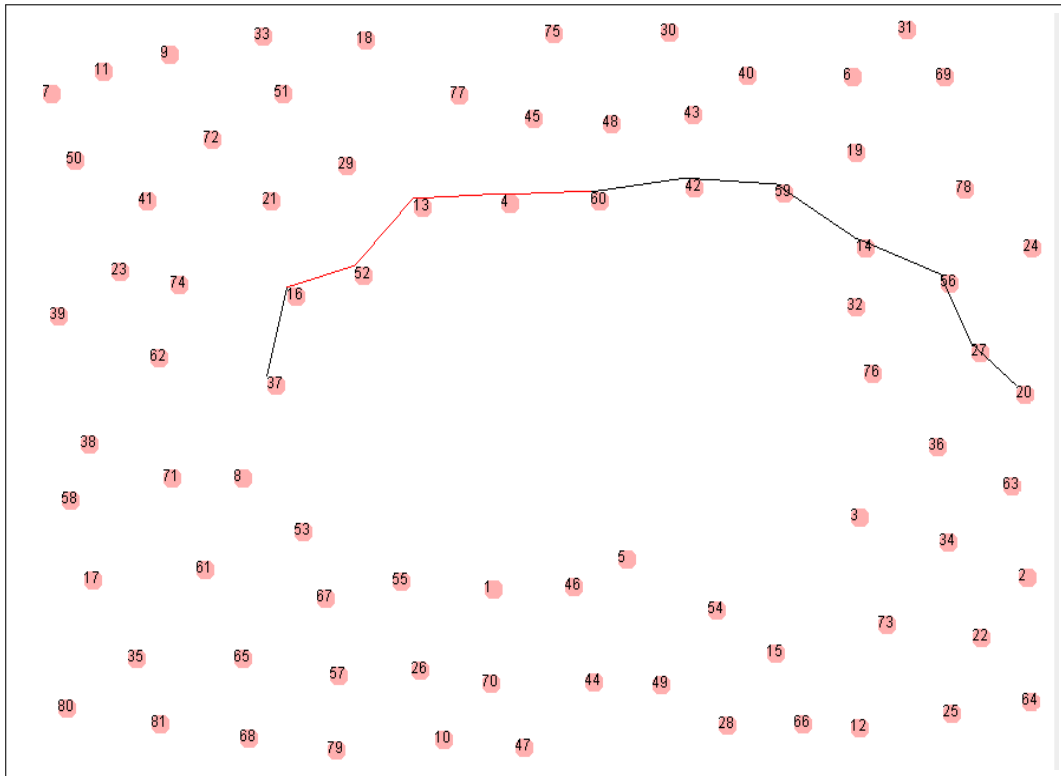
Here the screen shows the bypass routing the holes the correct path consists of different nodes with source and destination and each node has consume some energy which shown in graphically.



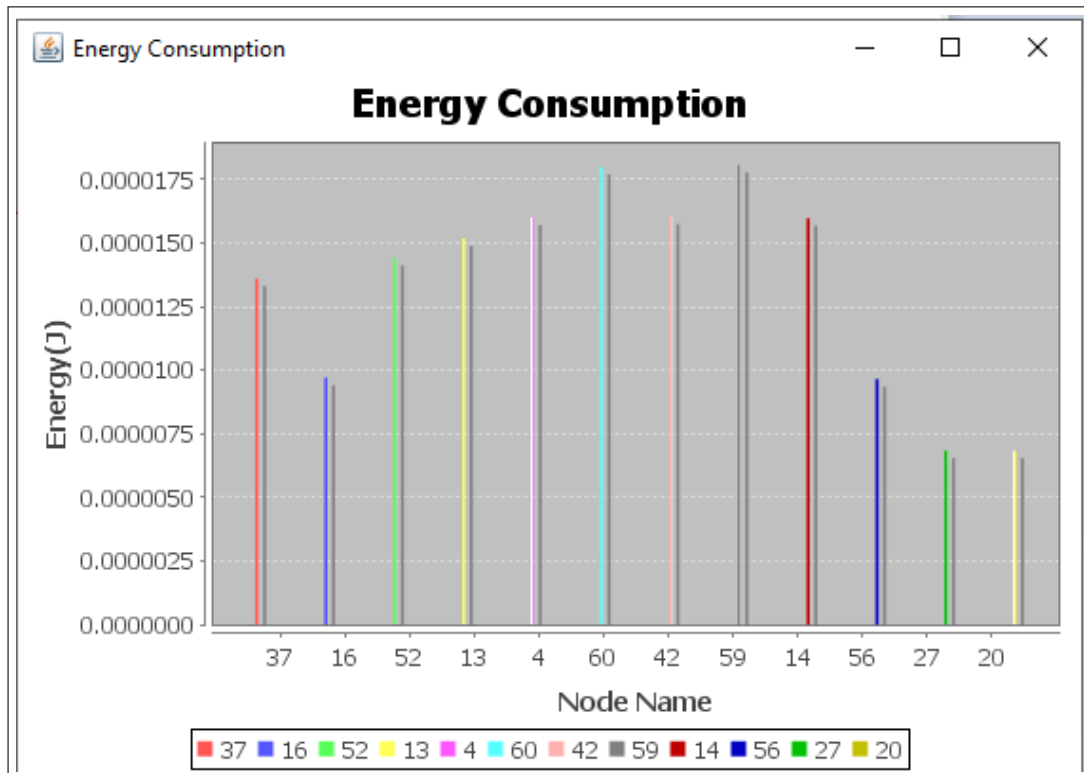
Here we have shown the simulation of path which bypass the routing holes. the source node and destination node indicate the starting and ending of network communication path



Overall energy Consumption and network density graphically represented



Another window of bypassing the routing holes in mobile WSN



Individual energy consumption by node after data dissemination.

## 10. TEST SPECIFICATION

### 10.1 PROJECT QUALITY AND TESTING REPORT

Testing is a process used to help identify the correctness, completeness and quality of developed computer software. Here testing activities are enlisted which have been carried out for the dissertation. It describes the software test environment for testing, identifies the tests to be performed and provides schedules for test activities.

#### 10.1.1 Goals and Objectives

1. To propose an efficient secure scheme for data markets, which simultaneously guarantees data truthfulness and privacy preservation.
2. To achieve the ultimate goal of the service provider in a data market is to maximize their profit.

#### 10.1.2 Testing Strategy

Tests come in several tests such as unit testing, validation of the smallest components of the system, and ensuring handling of no input and outputs correctly. Integration testing includes testing of an entire sub-system and ensuring that a set of components performs efficiently when united. Functional testing verifies end-to-end scenarios that the user will be absorbed in. The methodology for software testing is defined below.

#### 10.1.3 Unit Testing

A unit is the smallest testable part of the software. It usually has one or a few inputs and usually a single output. This design is for validation of the program that it functions properly so that it produces the desired output for the system. It is done on each module of the system. It is applied to each module of the system individually.

#### 10.1.4 GUI Testing

Functional testing (or GUI testing) governs how the user and the application interact with each other and also tests the application performance. This test includes how to display images. The proposed system is tested for user inputs against different modules, validations are done. The interface is tested for appearance of different controls, visibility graphs are tested. The GUI is tested whether each button is working properly or not. A GUI allows the user to interact with the system using the buttons. The test case is:

1. **Purpose of Test:** Respective procedure should be called and executed.
2. **Driver:** Each button on the form (type submit, browse, textArea)
3. **Stub:** Java swing component
4. **Test Procedure:** Click the specific button
5. **Expected Results:** On click of each button the respective procedure should be called and executed.
6. **Actual Results:** The functionality of the button is executed successfully.
7. **Pass/Fail:** Pass.

This testing involves following actions,

1. Check all elements for size, position, width, length and acceptance of characters or numbers. For

instance, you must be able to provide input to the input fields.

2. Overall functionality related to the performance of the user graphical interface is checked.
3. Check error messages are displayed correctly.
4. Check the all-result display is accurate or not.

### **10.1.5 Integration Testing**

Integration testing is the software testing process where individual units are combined and tested as a group. The purpose of this testing is to expose faults in the interaction between integrated units. Each individual component after unit testing is integrated and tested. Each time a new component is added to the system an integration test is performed. The entire system is tested as characterized by the extent of the advancement project or product.

It may incorporate tests based on risks and/or requirement specifications, business procedure, use cases and other abnormal states of system behavior, connections with the working frameworks, and system resources. The final test is used to verify that the system to be delivered meets the specification and its purpose.

### **10.1.6 Acceptance Testing**

It is performed with realistic and various image data for training to demonstrate that the software is working satisfactorily in testing phase. User acceptance is a critical phase of any project and requires significant participation by the end user. It ensures that system meets the functional requirements. Acceptance of end users is done by proper images output.

### **10.1.7 Validation Testing**

Validation testing will be done to ensure validation of the client's requirements given to the software team for the system compared to the performance and results from the software system. Given that the system has passed all the previous unit and integration testing, validating the system will measure an aspect of an implementation's behavior against an expectation and check whether the software complies with that expectation.

### **10.1.8 Performance Testing**

Performance testing is the testing, which is performed, to ascertain how the components of a system are performing, given a particular situation. In proposed system it is mainly focused on increasing the performance of system to increase the accuracy of the output and analysis of result is done by using accuracy and time parameters.

## **10.2 PURPOSE AND SCOPE OF DOCUMENT**

The purpose of this document is to present a detailed description of semantic trademark retrieval system. It will explain the purpose and features of the system, the interfaces of the system, what the system will do, the constraints under which it must operate and how the system will react to external stimuli. This document is intended for providing privacy and security to the users.

## **10.3 OVERVIEW OF RESPONSIBILITIES OF DEVELOPER**

1. Perform project design and development activities according to customer specifications.



2. Work with Manager in developing project plan, budget and schedule.
3. Coordinate with management in preparing project proposals and contractual documents.
4. Track project progress regularly and develop status reports to management.
5. Ensure that the project is completed within allotted budget and timelines.
6. Follow company policies and safety regulations for operational efficiency.
7. Research and recommend new technologies to carry out project development tasks.
8. Provide assistance to other Developers, perform peer reviews and provide feedback for improvements.
9. Develop cost reduction initiatives while maintaining quality and productivity

#### 10.4 TEST-CASES

Table 10.1: Test Cases

ID	Test Cases	Test Status	Test Result
1	Take the input as source and destination node	Successfully select the node	Test pass
2	Path selection using anchor list	Finalize the Anchor list and display	Test Fail
3	Data Dissemination with source and destination	Path establishes	Test Fail
4	Individual node Energy consumption	Calculated	Test Pass
5	Overall Energy consumption	Calculated	Test Pass

#### 11. APPLICATION

Although originally developed as a routing scheme that uses the physical positions of each node, geographic routing algorithms have also been applied to networks in which each node is associated with a point in a virtual space, unrelated to its physical position. The process of finding a set of virtual positions for the nodes of a network such that geographic routing using these positions is guaranteed to succeed is called greedy embedding. The following are some important areas which used in WSN are as follows:

1. Environment Monitoring
2. Industrial and Business Uses
3. In the Livestock
4. Habitat Monitoring
5. Forest Monitoring
6. Underwater Sensor Network

#### 12. CONCLUSION

We propose EMGR, a novel energy-aware multipath geographic routing protocol, to better recover the route from routing holes for WSNs. In EMGR, two node-disjoint anchor lists are provided to shift routing path for load balance, and the location information, residual energy and energy characteristics of nodes are employed to make routing decisions, thereby enabling an energy-aware multipath routing strategy.

### 13. REFERENCES

1. L. Cheng, J. Niu, J. Cao, S. Das, and Y. Gu, "QoS Aware Geographic Opportunistic Routing in Wireless Sensor Networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 25, no. 7, pp. 1864-1875, Jul. 2014.
2. C. Fraser, C. Kevin, S. Jose, and M. Sandra, "A Survey of Geographical Routing in Wireless Ad-Hoc Networks," *IEEE Commun. Surv. Tutor.*, vol. 15, no. 2, pp. 621-653, 2013.
3. S. S. Lan and C. Qian, "Geographic Routing in d-Dimensional Spaces with Guaranteed Delivery and Low Stretch," *IEEE/ACM Trans. Netw.*, vol. 21, no. 2, pp. 663-677, 2013.
4. B. Karp and H. Kung, "GPSR: Greedy Perimeter Stateless Routing for Wireless Networks," *Proc. ACM MobiCom 2000*, Boston, Massachusetts, USA, pp. 243-254, 2000.
5. D. Chen and P.K. Varshney, "A Survey of Void Handling Techniques for Geographic Routing in Wireless Networks," *IEEE Commun. Surv. Tutor.*, vol. 9, no. 1, pp. 50-67, 2007.
6. P. Bose, P. Morin, I. Stojmenovic, and J. Urrutia, "Routing with Guaranteed Delivery in Ad Hoc Wireless Networks," *ACM/Kluwer Wireless Netw.*, vol. 7, no. 6, pp. 609-616, 2001.
7. W. Liu and K. Feng, "Greedy Routing with Anti-void Traversal for Wireless Sensor Networks," *IEEE Trans. Mob. Comput.*, vol. 8, no. 7, pp. 910-922, 2009.
8. Q. Chen, S. S. Kanhere, and M. Hassan, "Adaptive Position Update for Geographic Routing in Mobile Ad Hoc Networks," *IEEE Trans. Mob. Comput.*, vol. 12, no. 3, pp. 489-501, Mar. 2013.
9. H. Frey and I. Stojmenovic, "On Delivery Guarantees of Face and Combined Greedy Face Routing in Ad Hoc and Sensor Networks," *Proc. ACM MobiCom 2006*, CA, USA, pp. 390-401, 2006.
10. Q. Fang, J. Gao, and L. J. Guibas, "Located and Bypassing Routing Holes in Sensor Networks," *Proc. IEEE INFOCOM 2004*, Hong Kong, China, pp. 2458-2468, Mar. 2004.
11. C. Petrioli, M. Nati, P. Casari, M. Zorzi, and S. Basagni, "ALBA-R: Load-balancing Geographic Routing Around Connectivity Holes in Wireless Sensor Networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 25, no. 3, pp. 529-539, Mar. 2014.
12. X. Xiang, X. Wang, and Z. Zhou, "Self-Adaptive On-Demand Geographic Routing for Mobile Ad Hoc Networks," *IEEE Trans. Mob. Comput.*, vol. 11, no. 9, pp. 1572-1586, Sept. 2012.
13. X. Li, J. Yang, A. Nayak, and I. Stojmenovic, "Localized Geographic Routing to a Mobile Sink with Guaranteed Delivery in Sensor Networks," *IEEE J. Selected Areas in Comm.*, vol. 30, no. 9, pp. 1719-1729, Oct. 2012.
14. S. Ruhropand I. Stojmenovic, "Optimizing Communication Overhead while Reducing Path Length in Beaconless Georouting with Guaranteed Delivery for Wireless Sensor Networks," *IEEE Trans. Comput.*, vol. 62, no. 12, pp. 2240-2253, Dec. 2013.
15. X. Wang, J. Wang, K. Lu, and Y. Xu, "GKAR: A Novel Geographic K-anycast Routing for Wireless Sensor Networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 24, no. 5, pp. 916-925, 2013.
16. A. Mostefaoui, M. Melkemi, and A. Boukerche, "Localized Routing Approach to Bypass Holes in Wireless Sensor Networks," *IEEE Trans. Comput.*, no. 63, no. 12, pp. 3053-3065, Dec. 2014.
17. G. Tan and A. M. Kermarrec, "Greedy Geographic Routing in Large-Scale Sensor Networks: A Minimum Network Decomposition Approach," *IEEE/ACM Trans. Netw.*, vol. 20, no. 3, pp. 864-877, Jun. 2012.
18. J. Sanchez and P. Ruiz, "Locally Optimal Source Routing for Energy-efficient Geographic Routing," *ACM/Kluwer Wireless Netw.*, vol. 15, no. 4, pp. 513-523, May 2009.