Data Hovering Algorithm for Improving Data Retention and Data Quality in Energy-Constrained Mobile Wireless Sensor Networks

Thesis by

Yingjie He

In Partial Fulfilment of the Requirements

for the Degree of

Doctor of Philosophy

Newcastle University

Newcastle upon Tyne, UK

2011

Acknowledgements

First of all, I would like to thank my supervisor Dr. Alan Tully who guided me through this research with numerous invaluable advices. I would also like to thank the other supervisors Dr. Paul Ezhilchelvan and Dr. Nick Cook for their support.

I wish to thank my parents, my family, and my wife Juan Qi for their love, support and encouragement.

I would also like to thank my friends Yishi Zhao, Huqiu Zhang, Xiao Chen, Wen Zeng, Wei Chen, and Mohamad Nazim Jambli for their assistances and making my life in PhD vigorous.

Abstract

A Wireless Sensor Network (WSN) is composed of numerous spatially distributed, low cost, low power and multifunctional sensor nodes which can be used to monitor the surrounding environment. In mobile networks, the sensed data collected by the sensor nodes may move out of the area where it has been gathered (area of origin) with its carrying node. A problem may arise in this situation: when requesting the historical information of a specific area, it is possible that none of the nodes currently located in such area can provide the required information. This thesis addresses the issue of retaining data it its area of origin in an energy-constrained, infrastructure-less mobile Wireless Sensor Network. The concept of this "Data Hovering" has been defined in which the location-based data hovers in its area of origin by transmission between network nodes. Based on this concept, several policies need to be defined as well as considering the constraints of WSN including limited energy and limited transmission bandwidth. The existing related work has then been investigated by examining how it proposed to define the Data Hovering policies, in order to explore the limitations. It has been found that the existing approaches are not well suited to mobile WSN, due to the unique characteristics of WSN. In this thesis, an autonomous Data Hovering algorithm consisting of defined policies has been designed to improve the data retention (data availability) and the quality of the retained data which ensures that the retained data represents different information. The defined Data Hovering algorithm has been implemented in a network simulator and a baseline with simple policies has also been selected in order to be compared with the defined policies. The evaluation in terms of data availability, data quality and energy consumption has then been carried out to analyze the behaviours of the defined algorithm. Finally, the potential future work has been suggested.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Introduction to Wireless Sensor Network

A Wireless Sensor Network (WSN) is composed of numerous spatially distributed, low cost, low power and multifunctional sensor nodes. Each sensor node consists of one or more sensing units with the ability to monitor the surrounding environmental conditions (e.g. temperature, light and sound), a processor and a radio transceiver, and it is usually battery-powered. These sensor nodes communicate with each other via a wireless channel, and they mainly use a broadcast communication paradigm [1]. These characteristics of sensor nodes enable the capability of WSN to be applied in a mobile environment and hence each sensor node does not need to be fixed at a specific location.

1.1.1 Sensor Node

The essential components of a sensor node in WSN consist of a sensing unit, a processing unit, a transceiver, and a power unit [1] (as shown in Figure 1.1 which has been extracted from [2]). A sensor node may also consist of other components to support additional features depending on different applications, such as location finding system, power generator and external memory [1, 3, 4]. In [5], Vieira et al. discussed the characteristics of the sensor node components in details.

- The sensing unit: A sensing unit is usually composed of a group of sensors and analogue-to-digital converters (ADCs). The sensors produce an electrical response based on the change of surrounding environment, and the ADC converts analogue signals produced by these sensors to digital signals and delivers them to the processing unit [1, 2, 5]. There are many types of sensors available, as for example temperature, light, humidity, pressure, motion, seismic, chemical and biological sensors [6]. The type of sensors being used in a sensor node is application dependent.

- The processing unit: The processing unit controls and collaborates with the other components of the sensor node to perform tasks and process data. A small storage unit is generally associated to the processing unit for local data processing and tasking [1]. Flash memory is widely used as this storage unit because of its low cost and the benefits of its storage capacity [4]. The most common processing unit in a sensor node is microcontroller because of its low cost and low power consumption, and alternatives include microprocessor and Field Programmable Gate Array (FPGA) [2]. Vieira et al. [5], discussed a comparison of different types of microcontrollers which are being used in the WSN.
- The transceiver: The transceiver in a sensor node enables the capability of communicating with other network nodes. It provides the functionalities of both transmitting and receiving data. The possible choices of communication media include Radio Frequency (RF), optical communication (laser) and infrared. RF is most commonly used in the WSN for general applications.
- The power unit: The sensor node is usually operating by the supply of batteries. The major power saving policies used in the WSN are Dynamic Power Management (DPM) [7] and Dynamic Voltage Scheduling (DVS) [8]. Current development of the sensor nodes enables the capability of harvesting energy from solar and vibration [6, 9].

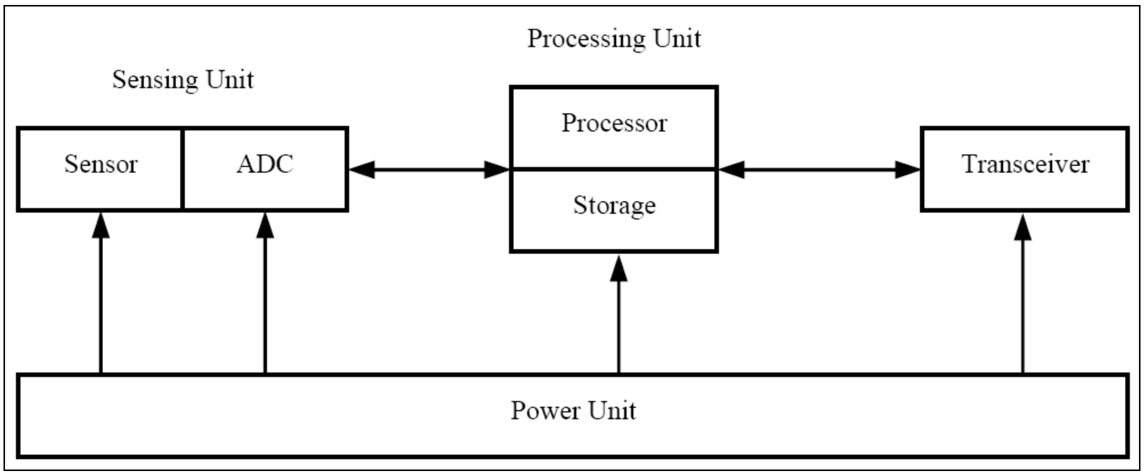


Figure 1.1 Sensor Node Components

Many commercial sensor nodes are available. A list of the commonly used sensor network hardware platforms is available in [10]. Below briefly discussed the examples of current sensor nodes.

At UC Berkeley, a family of sensor nodes has been released for general purpose WSN platform. MicaZ [11] is the latest product in Mica family developed by Berkeley. It is equipped with a low power ATmega128L microcontroller and a radio module with a data rate up to 250 Kbps. This sensor node allows several different sensor boards and data acquisition boards to be attached on top of the main processor board via a built-in standard 51-pin expansion connector.

Telos [12] (Figure 1.2) released later than the MicaZ with a set of new features: it is equipped with a new microcontroller (MSP430) to further reduce the power consumption, a built-in internal antenna, an extra on-board USB, a 64-bit MAC address for unique node identification, and integrated humidity, temperature and light sensors [6, 12]. Tmote Sky [13] was then released as the successor of Telos with enhanced performance, functionality and expansion.

The PicoRadio [9] project at Berkeley developed a radio transmitter – PicoBeacon, in which it is solely powered by solar and vibration [6, 14].

Medusa Mk-2 [15] and iBadge [16] have been developed at UCLA. These sensor nodes use more than one processor: one with high computational capability and another one will perform more signal processing tasks. In addition, iBadge is also equipped with a Bluetooth radio.

BTnode [17] features two independent communication modules: a Bluetooth radio and a low-power radio. These two radios can be operated simultaneously or be independently powered off.

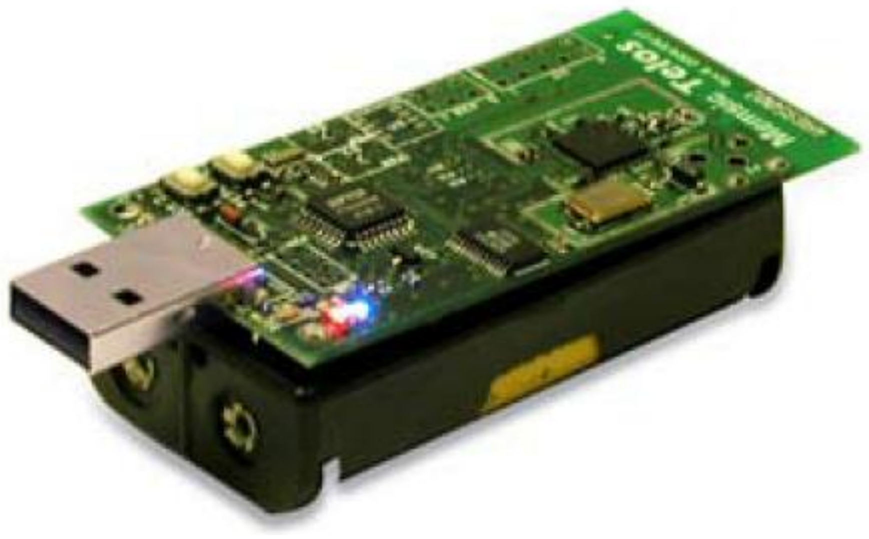


Figure 1.2 Sensor Node: Telos [12]

1.1.2 Applications

WSN has great potential for area monitoring and object tracking. It can take advantage of wireless connections rather than wired installation. It can be used in many application areas including but not limited to military target tracking, environmental sensing, habitat monitoring, traffic monitoring and control, health monitoring and inventory management. In military examples, sensor nodes can be placed in the battlefield to detect enemy intrusion [18]. In the area of environmental sensing, sensor nodes can be deployed to monitor the air pollution. For example, the carbon monoxide gas emitted from motor vehicles' exhaust, sulphur dioxide released from factories, and volcanic ash from a volcano eruption. The gathered data can be used for analysis or taking appropriate further actions based on the gathering data. WSNs can also be used to detect and predict forest fire by densely deploying sensor nodes in the forest to measure relative humidity, temperature, smoke, and wind speed and detect or predict a forest fire based on the collected sensed data [19]. In a landslide detection system [20], sensor nodes can provide the real time measurement of the movement of the soil in order to detect a landslide and alert the people during a landslide and possibly even before a landslide occurred. With traffic monitoring and control, sensor nodes can be used to monitor the traffic condition of a road segment. It is also possible to connect the sensor nodes to the traffic lights to regulate the state of the traffic light to give emergency vehicles higher priority (e.g. ambulance) [21]. For health monitoring, using WSN can improve the existing health care and patient monitoring [22]. For example, sensor node can be attached to an infant's clothing for detecting the sleeping position of an infant and alert the parents if such position would cause dangerous to the infant. Furthermore, sensor nodes attached to patients or emergency crews (e.g. fire fighter) can monitor their heart rate and blood pressure and appropriate actions can then be taken if such data exceeds a certain threshold. Moreover, sensor nodes can also be used to detect the environmental noise and inform the people within a particular range in order to avoid hearing impaired. In inventory management, sensor nodes with unique identification provide the location information of their attached products [23]. This unique identification can be used to query the product's details in a database. In contrast to the traditional methods of data acquisition and identification methods, by using pen-and-paper and barcode system, using WSN can improve the operational efficiency for tracking and finding products, and reduce the proneness of human errors.

The nodes in WSN are capable of monitoring the surrounding information and individually or collaboratively interact with the physical environment. Wireless communication and using batteries as the main power source enable the mobility characteristic of WSN. However, recharging the batteries might be difficult in some specific applications. Below discusses applications using mobile WSN in an energy-constraint environment.

- Underwater monitoring:

Deploying a sensor network underwater can monitor numerous conditions such as water temperature, water pressure, conductivity, turbidity and pollutants [24]. Sensor nodes can be placed on an Autonomous Underwater Vehicle which can move across the field to ensure necessary connectivity between the nodes, and collect data through collaboration with the other nodes.

- Underground environmental monitoring:

  WSN can be used in environmental monitoring in underground tunnel such as coal mining [25]. The sensor nodes can be carried by miner or other devices to monitor environmental factors including gas, water, dust and etc. to ensure the people can work in a safe environment. Underground WSN requires extra energy consideration, where sensor nodes are equipped with batteries, and it is difficult to recharge or replace batteries when the sensor nodes are operating underground.

- Search and rescue:

  WSN can also be used in search and rescue system [26] where it is comprised of mobile sensors worn by people. The location based information can be provided by these moving sensors. This information can be used to determine the location of a person which whom may be in an emergency situation.

- Habitat monitoring and animal migration tracking:

  In habitat monitoring, sensor nodes can be attached to the animals to track the movements and the living environments of the animals. The collected data can then be used for analysis [27].

1.2 Introduction to Data Hovering

1.2.1 Motivation

In a Wireless Sensor Network, the network nodes collect their surrounding information by sensing. The sensed data collected by a network node is location-based and it is associated with the time that it has been collected. If the collecting locations of two sensed data are spatially close to each other and the gathering time of these two data are also temporally close, then these two data are likely represent the same information. Thus, the sensed data is related to the certain area where it has been gathered, and this area is referred to as the "Area of Origin" of the sensed data.

However, in a mobile network, a sensed data may move out of its area of origin with its carrying node, thus it is getting lost from its area of origin. When requesting the historical sensed data of a particular area, the nodes that area currently located in such area may not be able to provide the required data.

1.2.2 Concept of Data Hovering

To retain the sensed data in its area of origin in a mobile WSN, one possible solution is to construct fixed infrastructure in the network to support data communication between network nodes and store the sensed data. The data can be kept in their area of origin by storing in the appropriate fixed infrastructure. However, the density of the required fixed infrastructure and its location depends on the size of the network area and communication range of the network nodes. Thus, in a large scale network area, a fixed infrastructure may not always be cost effective.

Another possible approach is to rely on data communication between network nodes without the aid of fixed infrastructure: the sensed data will be transmitted to other network nodes in the same area as the area of origin of this sensed data. Thus, the sensed data can be transmitted back to their area of origin even though the network nodes are moving. We call this approach "Data Hovering". The concept of Data Hovering was first introduced in [28], and they called it "Hovering Information".

The most typical application highlighting the situation of Data Hovering is traffic monitoring and control. Figure 1.3 shows a simple example of this application. The grey area in this figure represents a road segment which will be monitored. Each vehicle travelling on this road is equipped with a sensor node which monitors the surrounding environment, such as traffic information and road conditions. When a new vehicle (B) travels towards this road segment, it can receive the information of this road segment from the other vehicles (A) in order to avoid traffic congestion and accidents. However, if the vehicle (A) carrying the traffic data left this road segment, then this data may not be available to the new coming vehicles because it has also left this area with its carrying vehicle. Therefore, the data must be retained in the road segment where it has been collected.

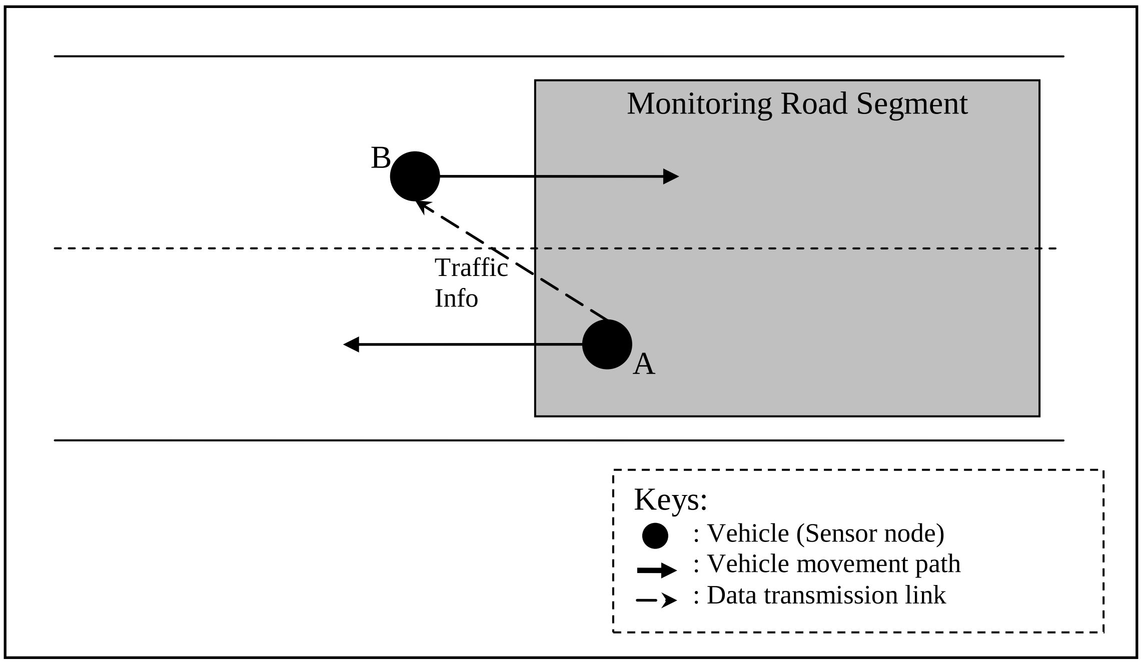


Figure 1.3 An example of applying Data Hovering in an application of traffic monitoring and control

1.2.3 Constraints and Data Hovering Policies

Wireless Sensor Networks are characterized by resource limitation constraints including limitation in power, processing, memory capacity and communication bandwidth [29]. Some of these constraints would have impact on defining Data Hovering policies:

C1. Limited power: Energy is the scarcest resource in WSN, because each network node is usually powered by batteries. When a node is running out of power, it will be disconnected from the network and its carrying data will be potentially lost.

C2. Limited memory: Each sensor node has a limited memory capacity, so it is not possible to store all the sensed data. Once the memory of a particular node is full, it cannot store more data either by sensing or receiving from other nodes.

C3. Limited bandwidth: The nodes in WSN communicate via a wireless medium, so they must share the capacity of the communication medium and this capacity is generally limited. In a mobile network, only partial data may be transmitted from a node to another because both bandwidth and time for data transmission between mobile nodes are limited.

Based on the scenario described in the Section 1.2.2, five policies would need to be defined to design a Data Hovering algorithm. In addition, the above constraints need to be addressed when defining Data Hovering policies. These policies are listed as follows:

P1. When to transmit (when should a node start and stop transmitting it data? For example, in Figure 1.3, when should node A start and stop transmitting its carrying data?)

P2. What to transmit (what data should be transmitted?)

P3. What to receive (what data should be received and locally stored?)

P4. When to delete (when should a node start deleting its data?)

P5. What to delete (what data can be deleted?)

The energy consumption for data transmission is much more expensive than local computation [30]. In an energy-constrained environment, there are two possible approaches with the capability to reduce the energy consumption caused by data transmission: reduce the total size of data to be transmitted, and reduce the power level for data transmission. Alternative method for reducing the size of transmitted data is to reduce the total number of transmission. This requires defining which data are most appropriate to be transmitted in policy P1 and P2, because it is not necessary to transmit all the data at all the time. Reducing the transmission power level will lead to higher data loss, since the communication range is also decreased. However, it is ideal to have the same level of data retained in their area of origin whilst the transmission power level is lower.

With limited memory, a data cleansing policy for destroying the sensed data on nodes must be specified in policies P4 and P5. Moreover, policy P3 should be defined to ensure that the network nodes will only store the appropriate received data.

Since only partial data can be transmitted to other nodes when the communication bandwidth is limited, it is possible that not all the data carried by a single node can be transmitted. Thus, it is necessary to define which set of data should be transmitted (P2). In particular, which data are more relevant, and only these set of data should be transmitted. In addition, a data prioritization should also be applied in policy P2, because it is possible that the limited bandwidth does not allow transmitting all the relevant data.

Table 1.1 summarizes which Data Hovering policies need to be defined to overcome the specific WSN constraint.

Constraints	**Policies**
C1	P1, P2
C2	P3, P4, P5
C3	P2

Table 1.1 The relationship between constraints and policies

1.3 Aims and Objectives

A piece of data is **available** if it has been retained in its area of origin. The **data availability** indicates how much data has been retained in its area of origin.

Due to the limited transmission bandwidth, only partial data could be transmitted to other nodes, so some data may not be retained in its area of origin. Thus, the quality of the retained data is another important factor for a Data Hovering algorithm. The quality of the retained data can be related to spatial or temporal diversity. As mentioned in Section 1.2.1, the data which were collected from the close locations and close times are likely to represent the same information. Thus, if these data have been transmitted, then less different information will be retained. In contrast, if the retained data are evenly spread over the network area or evenly spread over the data sampling time period, then the retained data will represent more information. The **spatial data quality** indicates how evenly the retained data is spread over the network area, and the **temporal data quality** indicates how evenly the retained data is spread over the data sampling time period.

The aim of this thesis is to design, implement and evaluate an autonomous Data Hovering algorithm to improve data availability and the quality of the retained data in an energy-constrained, infrastructure-less mobile Wireless Sensor Network. In particular, this thesis focuses on constraints C1 and C3 which have been described in Section 1.2.3; "when to transmit" and "what to transmit" policies will be defined to address the issues of limited energy and limited bandwidth.

The objectives for achieving this aim are:

- Investigate the policies of Data Hovering which need to be defined
- Examine the existing related approaches by looking at how they address the Data Hovering policies in order to explore the limitations and research gaps in the field of Data Hovering
- Design the Data Hovering algorithm to overcome the limitations of the existing related approaches and WSN constraints

- Implement the designed Data Hovering algorithm in a suitable network simulator
- Evaluate the performance of the designed Data Hovering algorithm by comparing with a selected baseline approach in terms of data availability, quality of the retained data, and the energy consumption.

1.4 Structure of this Thesis

The remainder of this thesis is structured as follows. Chapter 2 summarizes the related work and examines them by investigating how they proposed to defined the Data Hovering policies. Chapter 3 provides a detailed description of the proposed Data Hovering algorithm. Chapter 4 introduces the simulation settings and evaluation methodology. Chapter 5 presents an evaluation for the performances of the proposed Data Hovering algorithm. Chapter 6 summarizes the achievements of this work and discusses the potential future work.

Chapter 2

Related Work

2.1 Introduction

The concept of Data Hovering has been introduced in Chapter 1. This chapter will introduce the existing approaches in relevant areas, and examine the related approaches by looking at how they proposed to define the Data Hovering policies in order to explore the limitations and possible research gaps within the related approaches.

2.2 Query Processing

2.2.1 Relevant approaches of Query Processing

The concept of Data Hovering was motivated by the area of Query Processing in Wireless Sensor Network. In a Wireless Sensor Network, each sensor node is a separate data source which can generate information by sensing. It has been found that the aggregation of sensed data is more useful for user analysis than individual sensor readings. Therefore, it has been proposed to integrate sensor network and database technology to improve the management of data processed from sensor networks [31, 32], which is known as "Query Processing". Thus, a query of requesting the information of the sensor network can be answered by the sensed data.

Applying traditional database technology to wireless sensor networks is known as a "centralized approach". In this approach, the sensor nodes keep sending their sensed data to a powerful centralized database, and queries can be answered based on the existing data in this database [33]. This centralized approach is not generally applicable for wireless sensor networks, because sending all sensed data will consume a lot of energy. Since batteries are usually the main power source for sensor nodes, prolonging the network lifetime by reducing the energy consumption is one of the major aims of any sensor network.

In [34], the state-of-the-art of query processing in wireless sensor networks has been investigated by examining the existing approaches to explore the possible future research challenges.

Yao and Gehrke [33] proposed an enhanced centralized approach in which partial processing of the central database has been moved to network nodes. When a query has been issued, it will be injected to the network via a gateway node. An appropriate network node will be selected as a leader for generating the query results. Other non-leader nodes compute the partial results and send them to the leader, and the leader partially aggregates the results and sends back to the base station. Thus, only the required information will be extracted from the network.

Madden et al. [32, 35] view the whole sensor network as a distributed database. A routing tree will be constructed for data dissemination and collection of results. Only the required nodes that may be able provide the results will participate in processing the queries. Once a query has been issued, the query processor collects data from sensor nodes, filters it, aggregates it then routes it to the base station via an energy-efficient in-network processing algorithm.

The above query processors have been designed for static networks. In a mobile environment, the network topology will frequently change and the data loss rate is much higher. Thus, reconstructing the whole routing tree every time the network topology changes will consume much energy.

Huang et al. [36] proposed a walk-based query processing algorithm to overcome the issue of frequently changing network topology in mobile wireless sensor networks. In this algorithm, a routing tree will be constructed for each query. The root node of the tree is the query originator, visited nodes are internal nodes of the tree, and unvisited nodes are the leaves. Since the structure of this routing tree will likely be changed after a query has been issued, a repair mechanism has been introduced. When a node detects a broken link in the routing tree, it removes this link and so the tree is partitioned. It then attempts to find another node which is 1-hop from itself and it can link these two partitions. The tree will be repaired if the previous process succeeded, otherwise the node will continue searching.

Zhang et al. [37] proposed a buffering mechanism for managing the delivery of query results from mobile nodes to base station and the queries from base station to mobile nodes in order to overcome the issue of intermittent connectivity in mobile networks. In this mechanism, the local query processor continues to gather, store and process the collected data even during the periods with poor connectivity. After the connectivity resumes, the collected data will be sent in the order of perceived importance. In addition, a prioritization algorithm for managing query results has also been proposed.

In [38], Xu et al. propose a in-network query processing algorithm for mobile networks with highly dynamic topology. Unlike the query processing approaches for static networks, the query processing strategy of this proposed algorithm does not rely on routing structure. Instead, it uses a cooperative caching technique. In particular, nodes exchange their queries, results and sensed data with other nodes when travelling in the network. The queries may originate from a single node or multiple nodes. To overcome the constraints of limited bandwidth and limited memory, the queries and results are also prioritized for exchanging between nodes depending on number of demanding nodes and number of nodes which already have the results.

2.2.2 Limitations in Relevant Query Processing Approaches

In a mobile environment, when querying the historical data of a certain area, the nodes which are currently locating in that area may not be able to provide the answer, because the required data may not be fixed in their gathering area. Powerful network nodes which acting as central servers or buffers which placing in the appropriate locations in the network to store location-based data is one possible improvement, but it is too expensive. Queries could be answered by other nodes whose locations are not in the requested area, but this requires all network nodes to participate, so consuming much energy. In addition, it is possible that query results cannot be provided by the existing network nodes since the network nodes may also move out of the network area. Thus, the ideal solution would be retaining the location-based data in their area of origin. This limitation arises from the existing approaches in mobile wireless sensor network and it motivates the need for Data Hovering.

2.3 Data Replication and Hovering

2.3.1 Related Approaches

The algorithms of the following approaches were proposed for a mobile network whose network topology is frequently changed, so that no routing tree needs to be constructed and the data is associated with their relevant area.

The work by Shinohara in [39] proposed a family of data replication methods to improve the data availability in the network and balance the power consumption of nodes in a mobile ad hoc network (MANET). Unlike the meaning of data availability used in this thesis, this data availability indicates whether the data can be accessed by other nodes. To distinguish the meanings of these two terms, data accessibility will be used to represent the data availability of their proposed methods in the rest of this section. In MANET, some nodes may be disconnected from the network, due to no neighbouring nodes being located in their communication range. This results a network partition. The data which is currently carried by these nodes cannot be accessed by other nodes. Thus, the data accessibility will be lower, the more frequently network partitioning occurs. In addition, the node will consume more energy if the access frequencies of its carried data are higher. Replicating the data on other nodes would improve the data accessibility and balance the power consumption among nodes. Four different replication methods have been proposed:

- Expected Access (EA) – If a node requests a piece of data and it cannot find it in its local memory, then it request this data from another node. The replication procedure will then be triggered. The requesting node attempts to store the data in its own memory if its memory is not full. Otherwise, it replicates the data to other nodes. Data will be replicated to other nodes if they are frequently accessed by its carrying node and other nearby nodes, but with a small number of these nodes currently holding it.
- Weighted EA (WEA) – This method is based on the EA method and the criterion for selecting which data would be replicated has been changed. Since the node consumes more energy for accessing data which are held by other nodes than accessing its own data, this method considers the data access frequencies of its carrying node and other nodes separately. Different weights are predefined for different types of data access frequencies. The priority of data replication is based on their access frequencies and corresponding weights.
- WEA-Battery (WEA-B) – This method is based on the WEA method, in which the weight factor for access frequencies of nearby nodes dynamically changes based on a node's remaining battery power. When the remaining battery power at a node decreases, this node prefers to replicate the data which has a higher self access frequency. Thus, the data which was held by less power nodes can be acquired from other nodes which have the replica of such data, thus it prevents the nodes exhausting their battery power by transmitting the requested data. However, the data accessibility would also become lower if the remaining battery power of numerous nodes is low. This

is because all these nodes tend to only replicate the data which is frequently accessed on its own.

- WEA-Hop (WEA-H) – The path length between nodes is taken into consideration in this method, and it aims to prevent nodes from being accessed by other far away nodes.

Corbett and Cutting [40, 41] propose a location-based infrastructure-free annotation system. The system uses "virtual notes" (or data) which are generated by mobile devices to represent the information relevant to specific area, and these data can be retrieved by other nodes by querying the existing nodes in that area. The size of the relevant area of data can be increased depending on the locations of interested nodes. The location-based data is held in the relevant area by broadcasting to neighbouring nodes. Four replication policies, including "when to broadcast" and "what to broadcast", have been defined in order to improve the availability of data in the area of relevance and minimize the number of messages transmitted between mobile nodes.

- Publish – A piece of data will be immediately broadcast when it has been generated.
- Periodic – Each node periodically broadcasts its data in fixed time interval. The transmission priority of data is based on the time that this it has been locally stored: data which is recently received has lower transmission priority.
- Location-aware Periodic – The data whose area of relevance is at least partially within the communication range will be periodically broadcast.
- All – This policy combines the Publish and Location-aware Periodic policies, so that data will be transmitted after it has been generated, and periodically retransmitted if its area of relevance is overlapped with the communication range of its carrying node.

The aim of the work proposed by Leontiadis and Mascolo [42] is to disseminate traffic information around a specific region of a hybrid network consisting of fixed info-stations and moving vehicles. Their proposed system works in a publish/subscribe way: the info-stations publish some traffic information, and the vehicle drivers subscribe to the information of interest. Once the information has been published, a small number of vehicles will be selected as replica owners to carry such information, and they will act as mobile info-stations to periodically broadcast the information in order to disseminate the information to all vehicles in the area of interest. Because there are only a certain number of vehicles selected as replica owner, the information will be moved out of the interested area with its owner so that the information will be lost. Thus, the major objective of their system is to keep the data alive in the relevant area for a certain amount of time. They have proposed two algorithms: next replica carrier selection to select next carrier of the traffic information when the current replica owner is moving out of the relevant area, and number of replicas to estimate the appropriate total number of replica owners required to disseminate the information to all vehicles which are interested in the published information. The next carrier of replica owner will be selected based on the directions and locations of the neighbours of current replica owner: the current owner groups the neighbours into cluster, and randomly selects one vehicle from the cluster with most uninformed subscribers to be next carrier. To ensure there are an appropriate number of replica owners in the area of interest, the system merge useless replicas with other replicas and creates more replicas in the area where there are a large number of uninformed subscribers.

The Hovering Information approach of Castro et al. [43] aims to retain the hovering information in its belonging area of a network consisting of only mobile nodes. A piece of hovering information is a geo-localized data which has been predefined. This hovering information is responsible to keep itself available in its associated area by storing in mobile nodes whose locations are inside the area which is the same as its associated area. To satisfy this requirement, hovering information uses mechanisms including active hopping, replication and dissemination among mobile nodes without the assistance of any fixed infrastructure. The concept of areas of which hovering information is associated (as shown in Figure 2.1 [43]) have been introduced in this approach to assist the mobile nodes to decide their behaviours depending on different locations. The radii of these areas can be set by user.

- "Anchor area": The grey area in Figure 2.1 indicates the anchor area where the hovering information should be retained. The anchor area is a circular area whose centre is at the location of the information which is called the "anchor location". Each piece of information has its own anchor area.
- "Safe area": The information will not be transmitted when its carrying node is currently located in its safe area.
- "Risk area": The "risk area" is a ring centred at the anchor location, which overlaps with the anchor area and is limited by the safe area. The information will be transmitted when its carrying node is in the risk area of such information. The "risk radius" indicates the distance between the anchor location and the periphery of the risk area.
- "Relevant area": The "relevant area" is a circular area whose radius is bigger than the radius of the risk area. Data will not be transmitted by its carrying node whose location is inside the area between the risk area and the limit of the relevant area of this data.
- "Irrelevant area": The area which is outside relevant areas is considered to be the irrelevant area. The information will be removed from the memory of its carrier when its carrier is currently located in the irrelevant area of such information.

Each mobile node periodically checks its current location and computes the distance between its current location and the anchor location of each stored data. If this distance is greater than the radius of the safe area and less than the risk radius of a data, then the data's carrying node is in its risk area. When the data's carrier is in its risk area, this data will be periodically transmitted with the same time interval as the location checking interval. Two replication algorithms have been proposed: "Broadcast" and "Attractor Point" algorithms. In the Broadcast algorithm, a node broadcasts the data to all its neighbouring nodes. Unlike the Broadcast algorithm, the data will not be transmitted to all nodes within the communication range of its carrier in the Attractor Point algorithm. Instead, the node attempts to transmit the relevant data to the neighbouring nodes whose locations are closer to the anchor location of this relevant data. This has been done by computing the distances between the anchor location of the data which will be transmitted and the current location of each neighbouring node, and ordering them.

Caching and cleaning policies have also been defined in this approach. When a node receives a new piece of data from its neighbours, it stores the data in its memory if duplicate copy does not exist in its memory. When the distance between the node's current location and the anchor location of data which is stored in this node is greater than the radius of the relevant area, this data will be removed from this node's memory.

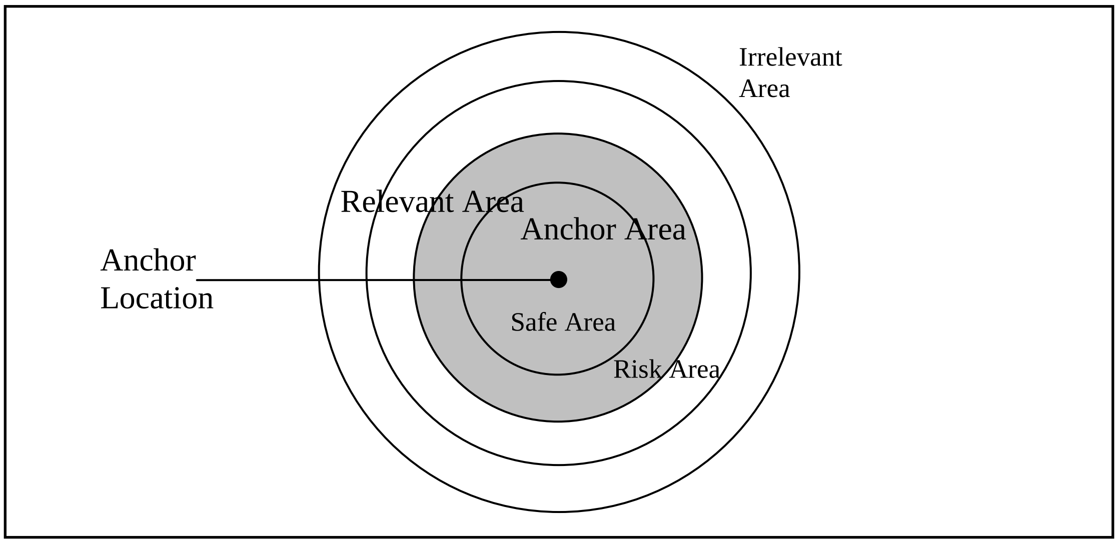


Figure 2.1 Concept of Areas in Hovering Information

A data replication algorithm which extends the Broadcast algorithm of the Hovering Information approach has been proposed by Fernandez-Marquez et al. in [44]. This approach aims to spread the predefined location-based data over its area of origin whilst keeping a minimum number of replicas. To achieve this aim, a "Broadcast with Repulsion Replication" algorithm has been proposed. This algorithm simplifies the concept of areas in Hovering Information, and there is only one area associated with the data, which is the anchor area. Data can be removed from the memory of its carrier when the current location of its carrier is outside its anchor area. If a node is inside an anchor area of a particular data, two different policies can be applied: broadcast and repulsion. If this node finds another neighbouring node which is within its communication range has another copy of the data (D) related to current anchor area, then the repulsion is triggered. Otherwise, the broadcast is triggered, so that the data will be broadcast to all neighbouring nodes within the communication range of its carrier. In repulsion policy, a desired location of the data will be calculated. This desired location is calculated based on the locations of all nodes which storing the data which is the same as the data D and their locations are within the communication range of the carrier of D. If the current carrier of D has the closest distance to desired position, the data D will be retained on its current carrier. Otherwise, the data D will be transmitted to the node which has the closest distance to the desire location.

The Floating Content by Kangasharju et al. [45] propose a data dissemination and management algorithm in a network consisting of mobile nodes. The aim of the proposed dissemination algorithm aims to distribute the data to mobile nodes in its area of origin. In this dissemination algorithm, a piece of data will be replicated to another node if the carrier of this data met another node in its communication range and another copy of this data does not exist in the memory of another node. To avoid the unlimited distribution of the data, a data management mechanism will be used to prioritize the data for transmission and storage. This prioritizing decision will be made based on the size of the area of origin of the data and the distance from its created location. Thus, when a node needs to replicate or store a piece of data, the data with smaller size of area of origin and shorter distance between current location of its carrier and its created location have the higher priority. This results that the probability of a node carrying data at the location closer to the data's created location is higher; this probability decreases when the carrying node of the data is farther away.

Xeros et al. [46] proposed four policies to disseminate the data in the data's relevant geographical area of a Vehicular Adhoc Network (VANET). In their proposed scenario, there is at least one node in a geographical area (Hovering Area) that is responsible for storing or generating data related to this area. The aim of the proposed policies is to disseminate the generated data to all nodes in hovering area whilst minimizing the volume of exchanged data.

- “Blind flooding” – Each node broadcasts the data whenever it finds another uninformed node.
- “Sender and receiver in area”: The data will be exchanged if both sender and receiver locate in the hovering area, and in the communication range of each other.
- “Receiver in area”: The data will be exchanged between sender and receiver if and only if the receiver is in the hovering area, and both are in the communication range of each other.
- “Probabilistic Flooding” – This policy is based on “receiver in area” policy in which the data will be transmitted to the nodes whose locations are inside the hovering area. In addition, when an informed node finds another uninformed node which is outside the hovering area, it decides whether to transmit the data based on a probability. This probability is calculated by either a strictly decreasing step function or a Gaussian like function. The input variable for these probability functions is the distance between the location of the uninformed node and the hovering area. This will ensure that the transmission probability decreases when the distance between uniformed node and the hovering area increases.

2.3.2 Limitations

In the existing approaches, each predefined data has its own area of origin with a predefined size. However, this is not well suited to wireless sensor networks. Sensor nodes are usually densely deployed in the network area for collecting location-based data. Each sensor node will take a sampling data of the surrounding information in a certain time interval. Thus, numerous sensed data will be generated, and the areas of origin of these data are likely to be overlapped. The node carrying these data needs to compare its current location with the area of origin of each data to determine whether a data should be transmitted. In this case, it is possible that this node may have to keep transmitting its carrying data when it is moving in the network area because it is moving out of the area of origin of every data. Since the data collected from the close locations and close times are likely to represent the same information, some subset of the transmitted data may represent the same information. Therefore, it is possible to use one appropriate area to represent the areas of origin of the data whose locations are spatially close.

Moreover, only partial data might be transmitted to other nodes due to the limited communication bandwidth of the sensor network. If most of the transmitted data represent the same information, then the quality of the data retaining in their area of origin is low. Thus, it is necessary to define a prioritization policy for ordering the data which will be transmitted in order to ensure different information has been retained.

The next carrier selection of the data has been defined in some related approaches, so that the data will be transmitted to the appropriate nodes by multicasting. This can be well suited in a small scale network. However, in a large scale network, the global identifications may not be assigned to the sensor nodes due to large number nodes, and broadcasting is usually used in wireless sensor network. Once a piece of data has been broadcast, all the neighbouring nodes can hear such data. Thus, the "what to receive" policy would need to be defined for this purpose. In particular, nodes decide which data needs to be stored in its memory when receiving a new piece of data.

Chapter 3

Data Hovering Algorithm

3.1 Overview

The related approaches have been introduced in Chapter 2. This chapter introduces the proposed data hovering algorithm, which is composed of when to transmit, what to transmit and data prioritization policies. The "when to transmit" policy defines when a network node should start and stop transmitting data; the "what to transmit" policy defines which data should be transmitted when the start or stop transmitting event triggered. The data prioritization policy defines the transmission order of data of a single node.

These data hovering policies are designed for a network consisting of numerous sensor nodes which are moving inside this network area, where there is no fixed infrastructure constructed in the network to support information exchange.

The network area is formed by several squared areas as shown in Figure 3.1, and each single square is called a "Grid". These Grids can have different spatial granularities. Inside the region which contains more Grids with smaller spatial granularities, more data would be generated and more nodes would travel inside. For example, when monitoring the traffic of a city, the central region in Figure 3.1 which consisting of more Grids with smaller size would represent the city centre which has more vehicles travelling inside.

The squared area of origin will be used in this defined Data Hovering algorithm, instead of circular area of origin which has been used in existing related approaches, introduced in Section 2.3.1. By using a circular area of origin, transmission of a particular data will be started when the carrier of such data is moving out of its area of origin. This requires periodically computation of the distance between the current location of the carrier and the sampling location of the data in order to determine whether this distance is greater than the radius of the circular area of origin of the data or not. This calculation of distance includes the arithmetic operations of square and square root which would results higher computational complexity. In case of using squared area of origins, it is not necessary to compute the distance. Instead, this only requires calculating which Grid a node is currently located in by truncating the coordinates of its current location (will be introduced in Section 3.4.1). Thus, it reduces the computational complexity.

A single Grid is thought to be the area of origin of a particular data, if this data's sampling location is inside such grid. Instead of using one area of origin for each data like the existing related approaches introduced in Section 2.3.1, each single Grid can represent the area of origin of more than one data. The reason for this has been discussed in Section 2.3.2. In addition, an appropriate size of the Grid should be specified for different applications in order to ensure that data which were taken from the same area of origin and close sampling time represent the same information.

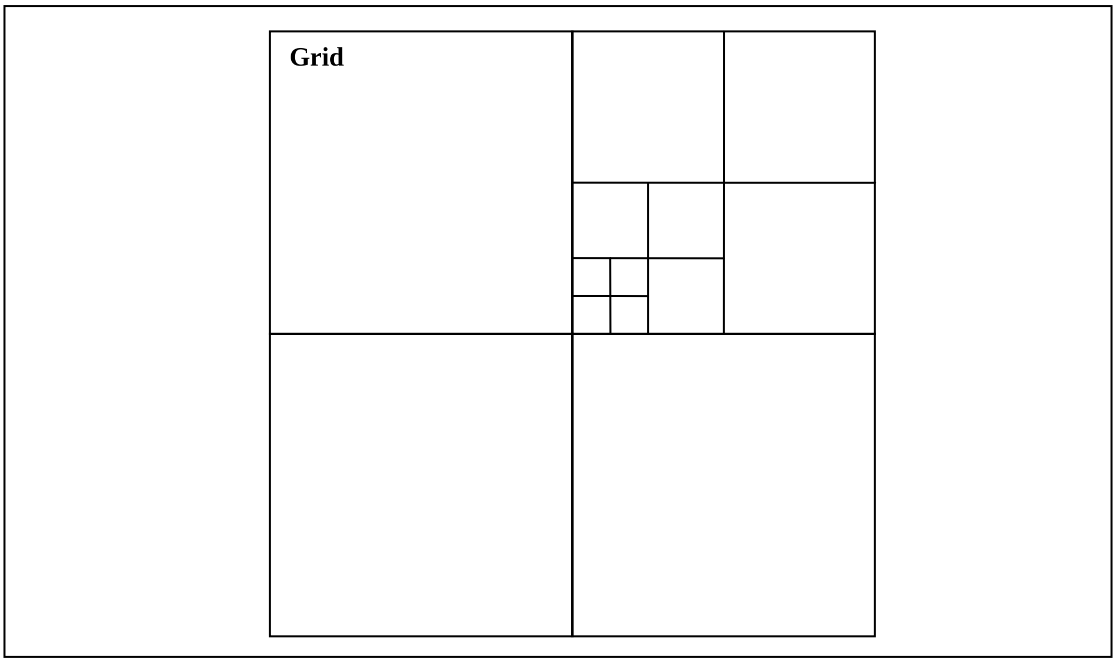


Figure 3.1 Grids with different spatial granularities

3.2 Assumptions

The following assumptions have been made in order to make the problem simpler and only focus on improving data availability and data quality.

- Limited energy: Network nodes are powered by batteries. When a node's power level is low, it will be disconnected from the network.
- Limited time to transmit data: The network nodes do not have enough time to transmit all their carried data during the transmission period. This will be affected by network bandwidth, node's moving speed, and total size of node's carrying data.
- Location awareness: Each network node has ability to know its current location, for instance, by using a built-in GPS or triangulation from nodes with GPS.

3.3 The Baseline

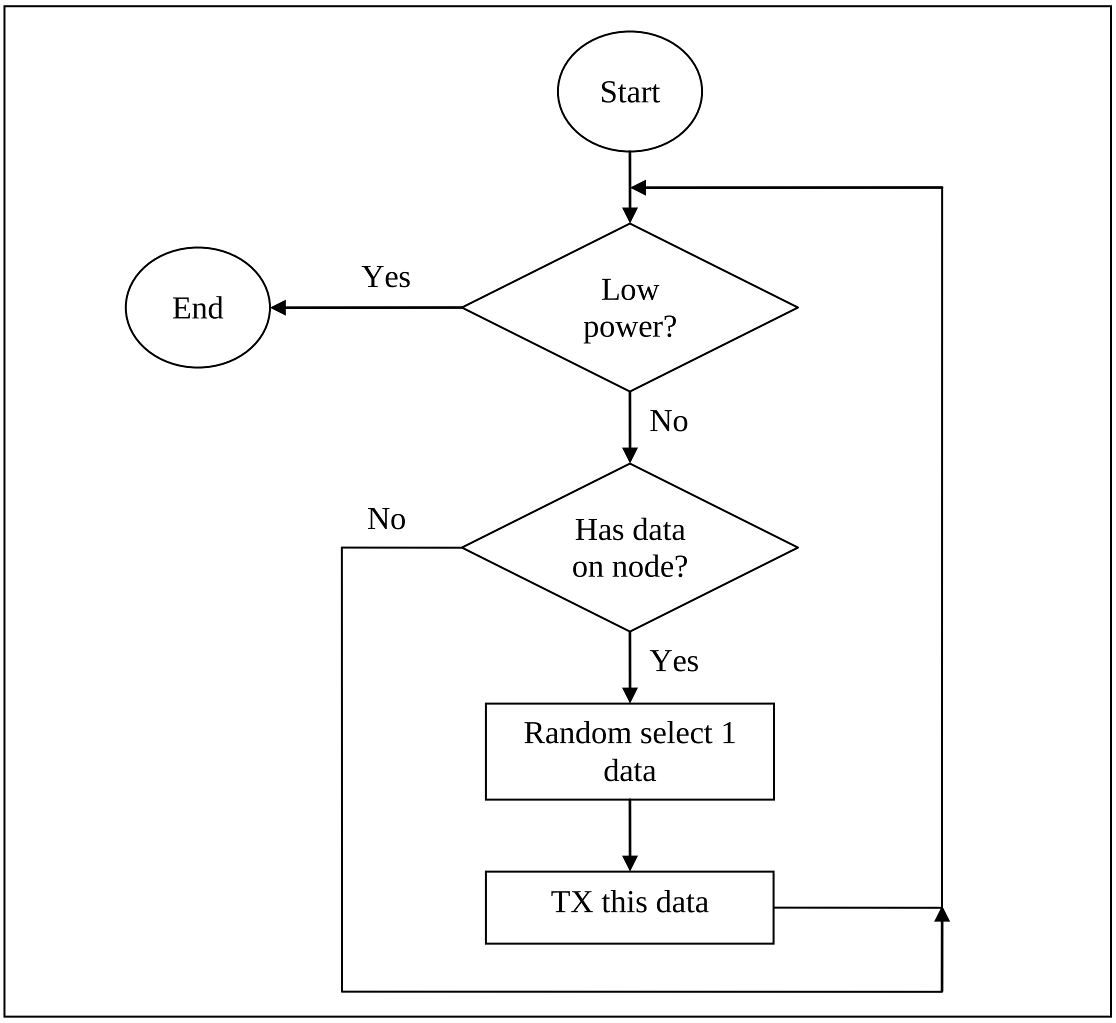


Figure 3.2 Flow chart of the Baseline

Before introducing the defined policies of data hovering algorithm, the baseline approach which is used to compare the performance with the proposed data hovering algorithm will be introduced.

When a node is moving in the network area, (Figure 3.2) it checks to see if it has any data, then it randomly selects one piece of data and transmits it; otherwise transmission of data will not be started until it carries at least one piece of data, thus receiving a new piece of data from another node. After the process of data transmission completed, the node will then trigger another random data transmission. This process of transmitting the random data will not stop until the node’s power level is low.

When a node received a new data from another node, it checks whether it already has the same data by comparing their sampling locations and sampling time. If both of them are the same, then the same data has been already been stored, so the new received data will be discarded; otherwise, it stores the received data.

3.4 When to Transmit policy of the Data Hovering algorithm

3.4.1 Defined Terms

Grid – G(X, Y, Adjacent_Grids, GLength)

- X, Y: the location of top-left corner of this grid. For example, in Figure 3.3, the location of Grid G1 is (X_1, Y_1).
- Adjacent_Grids: a list of grids which are adjacent to this grid. For example, in Figure 3.3, G2, G4 and G5 are the adjacent grids of G1.
- GLength: the length of each side of this grid.

Node – N(x, y, X, Y, preX, preY, Tx_Grids)

- x, y: represents the node’s current location.
- X, Y: indicates which grid that this node is currently in. This (X, Y) is periodically calculated by truncating the node’s current location (x, y), and it must be matched one of the grid’s (X, Y). To calculate the node’s current truncated location, the node’s current x coordinate divides by the length of the grid, and the quotient will be rounded to integer by removing the decimals, and then this result multiplies the length of the grid to calculate the X coordinate of

the node's truncated location; the same calculation which replacing the node's current x coordinate with the node's current y coordinate will be applied in order to calculate the Y coordinate of the node's truncated location. (In practice, this divide, truncate and multiply could be implemented by simply setting lower order digits to zero.) For example, in Figure 3.3, the initial (X, Y) of the node is (X_4, X_5) which is the same as the location of G4.

- preX, preY: represents the last grid that this node was in. This pair of preX and preY is the same as the location of one existing grid.
- "Relevant Grids" – Tx_Grids: a list of grids that each grid in this list is the adjacent grid of the current grid, and it has been previously entered by this node and only if this node has not left the grid's adjacent grids in its path. (Section 3.4.4 provides an example of how the contents of relevant Grids changes when a node moves through the Grids.)

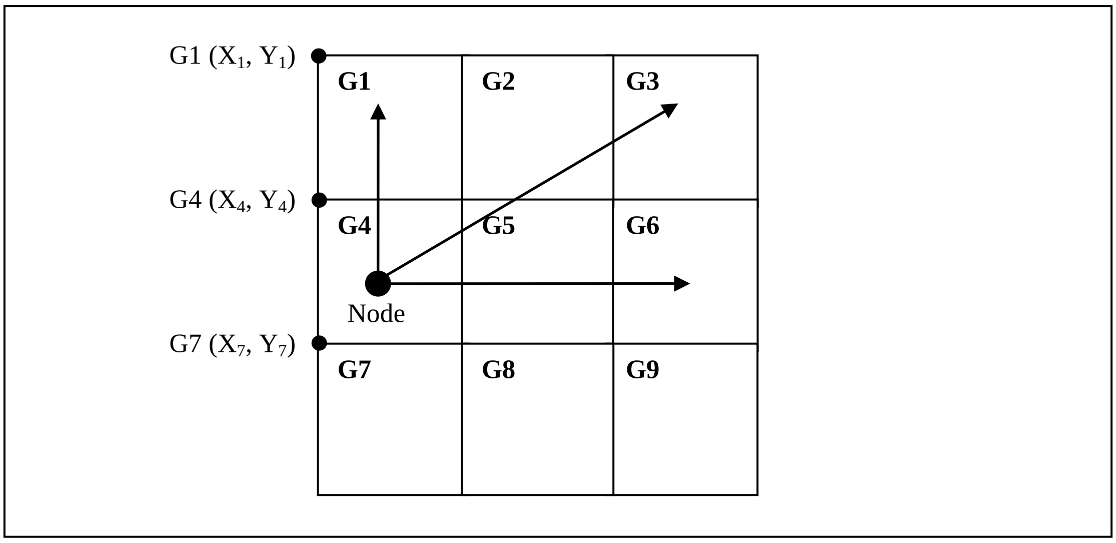


Figure 3.3 The network node moving through the Grids of a network

3.4.2 When to Start Transmission

The node periodically checks its current location (x, y), and it computes which Grid (X, Y) that it is currently located. It then compares X with preX and Y with preY, if either of them is different, then this node has left its current Grid. It sets its "Relevant Grids" (Tx_Grids) by adding its previous leaving Grid (preX, preY) to the end of Tx_Grids, and removing the Grids from Tx_Grids which are not adjacent to this node's current Grid. (An example of how the relevant Grids will be changed when a node's current grid is changing is described in section 3.4.4.) Transmission of data will then starts, and the previous Grid sets to the current Grid, thus preX = X, and preY = Y.

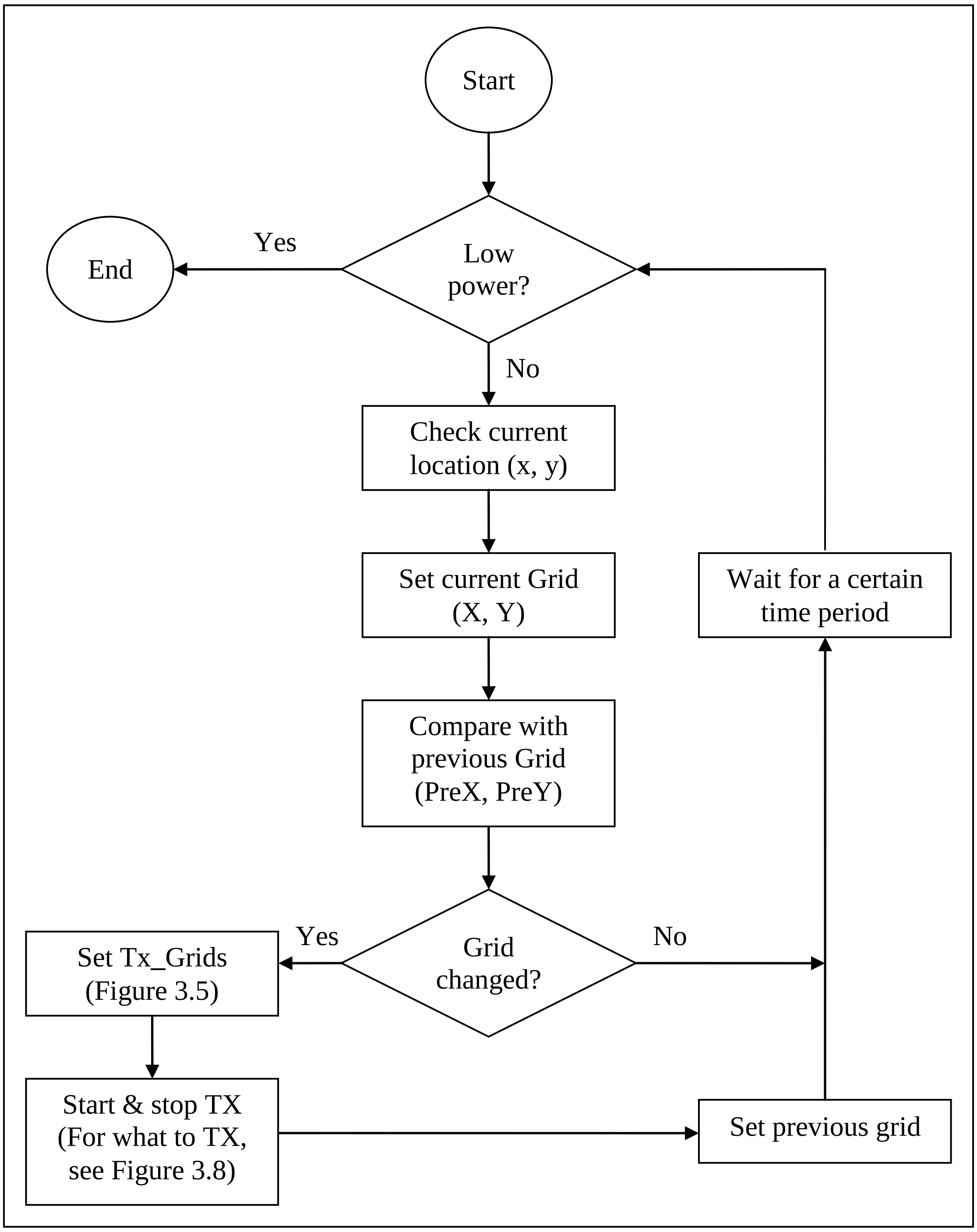

Start
Low power?
Yes
End
No
Check current location (x, y)
Set current Grid (X, Y)
Compare with previous Grid (PreX, PreY)
Grid changed?
Yes
No
Set Tx_Grids (Figure 3.5)
Start & stop TX (For what to TX, see Figure 3.8)
Set previous grid
Wait for a certain time period

Figure 3.4 Flow chart of when to transmit policy for a single node

3.4.3 When to Stop Transmission

Transmission of data related to G(X, Y) continues until the node is not in any of the adjacent Grid of G(X, Y) by removing the Grid from Tx_Grids. (Figure 3.4 shows the flow chart of when to transmit policy including when to start transmission and when to stop transmission.)

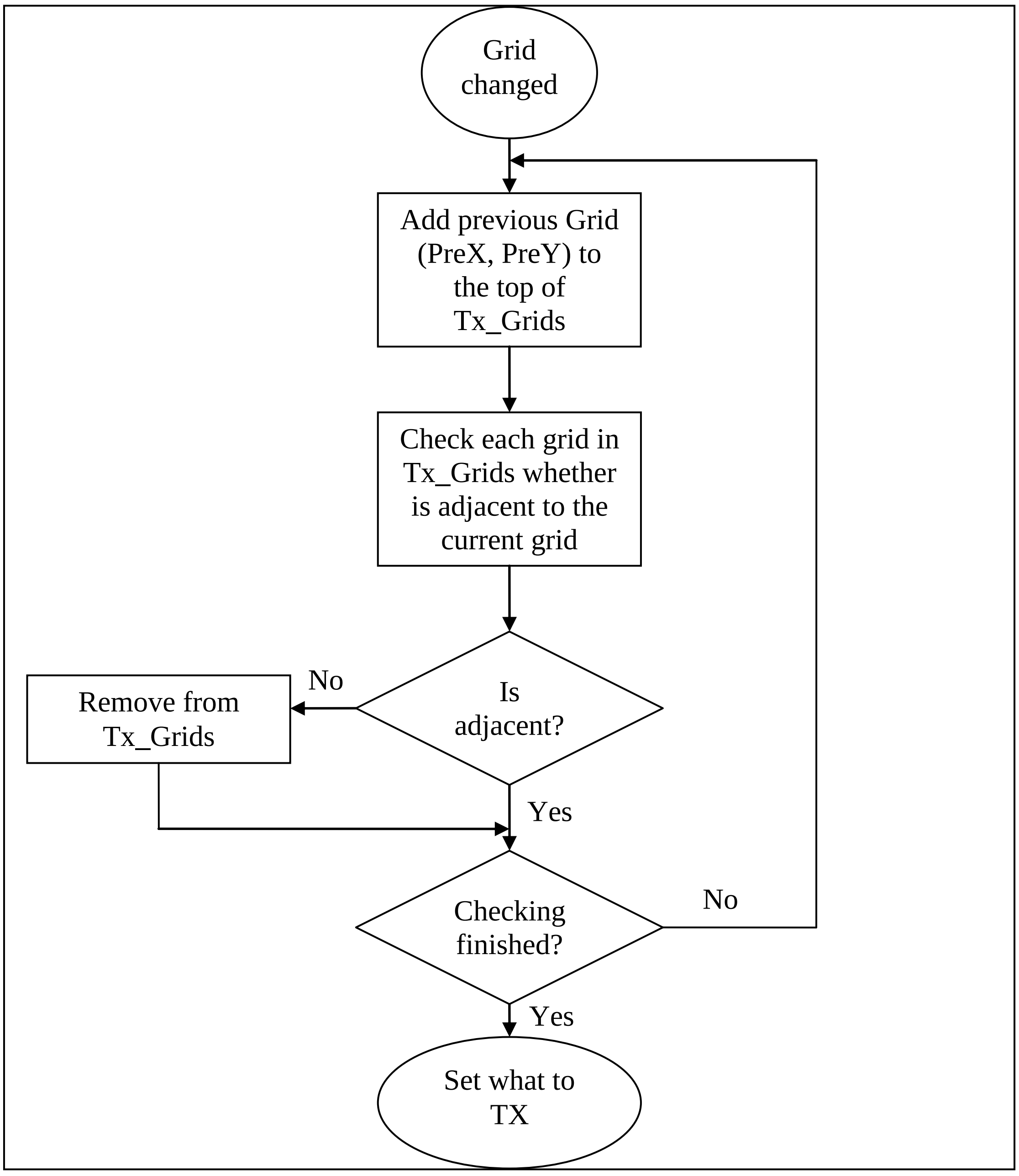

Grid changed
Add previous Grid (PreX, PreY) to the top of Tx_Grids
Check each grid in Tx_Grids whether is adjacent to the current grid
Is adjacent?
No
Remove from Tx_Grids
Yes
Checking finished?
No
Yes
Set what to TX

Figure 3.5 Flow chart of setting Relevant Grids

3.4.4 Example of Relevant Grids

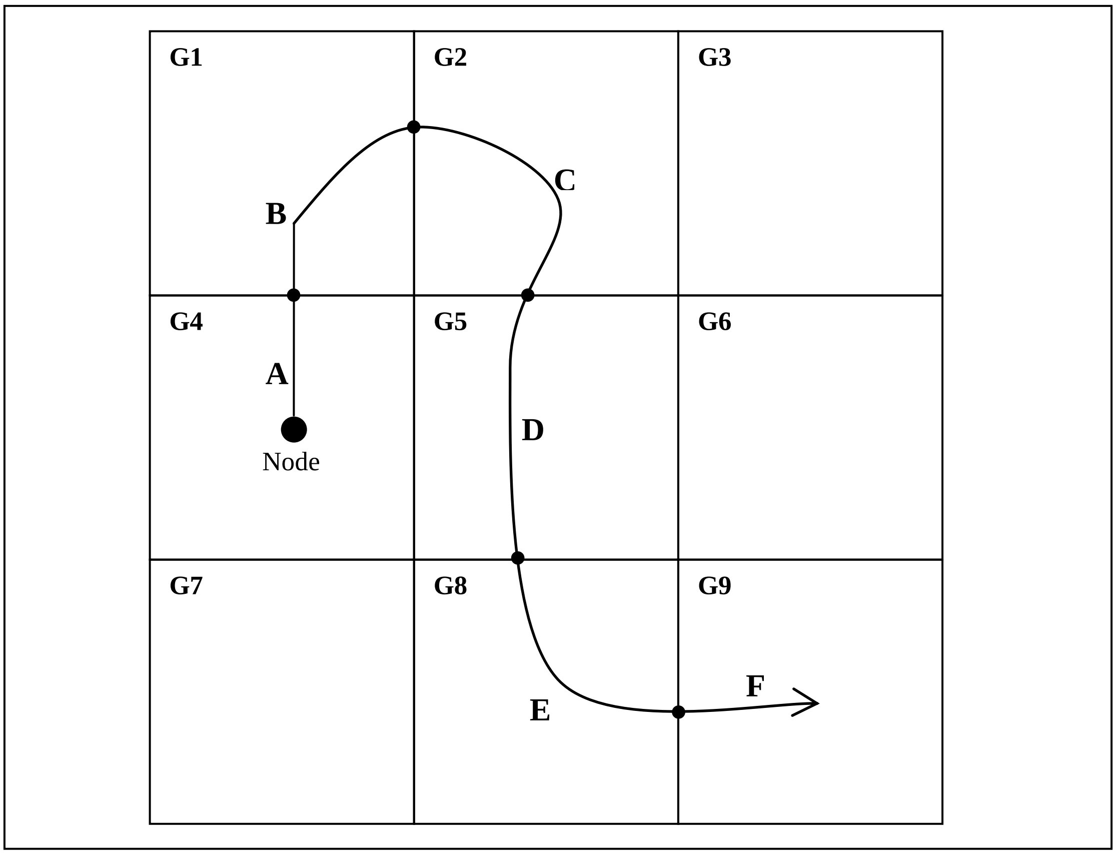


Figure 3.6 Example of how the relevant grids will be changed when node's current grid changed

This example shows how the Relevant Grids (Tx_Grids) will be changed based on the change of the current grid of the node. A node is initially deployed at a location in Grid (G4), and it moves to a location in G9 via G1, G2, G5 and G8 with the sequence illustrated in Figure 3.6. The moving path of this node can be divided into 6 segments (A – F) based on the changes of its current Grid. The contents of Relevant Grids will be changed as shown in the following table (Table 3.1):

Path Segment	**Tx_Grids**	**Description**
A	Empty	The Tx_Grids is initially empty, before the changing of node’s Grid.
B	G4	When the node moved into G1, it added the last Grid that it was in which is G4 to the end of Tx_Grids.
C	G4-G1	When the node moved into G2, it added its last entering Grid which is G1 to the end of Tx_Grids.
D	G4-G1-G2	When the node moved into G5, it added G2 to the end of Tx_Grids.
E	G4-G5	When the node moved into G8, it added G5 to the end of Tx_Grids. It then removed G1 and G2 from the Tx_Grids, since they are no longer the adjacent Grids of the current Grid.
F	G5-G8	When the node moved into G9, it added G8 to the end of Tx_Grids, and removed the non-adjacent Grid G4.

Table 3.1 The change of contents in Relevant Grids

3.5 What to Transmit policy of the Data Hovering algorithm

3.5.1 Data Organization

For the purpose of efficiently selecting which data should be transmitted, the organization of data plays an important role. As the data’s truncated sampling location determines which grid it belongs to, the data storing in a particular node are grouping based on their truncated location (as shown in Figure 3.7), thus the data with same truncated location will be stored in the same group. Each group is called **Grid_Data**, which has a unique identification (GX, GY) to distinguish from others, where (GX, GY) is the same as one of the network grids (X, Y). When a new data has received, it will be inserted into the appropriate Grid_Data where its truncated sampling location is the same as the Grid_Data’s (GX, GY).

Data in a node	
Grid_data G1 (GX_1, GY_1)	**Grid_data G2 (GX_2, GY_2)**
Prioritized data related to G1	Prioritized data related to G2
Grid_data G3 (GX_3, GY_3)	**Grid_data G4 (GX_4, GY_4)**
Prioritized data related to G3	Prioritized data related to G4

Figure 3.7 Data organization in a single node

3.5.2 Defined Terms

Data – D(DX, DY)

- X, Y: represents the grid that this data belongs to. This (DX, DY) is calculated by truncating the data's sampling location, when it was being collected. This (DX, DY) must be the same as one of the existing Grids (X, Y).

Node – N(X, Y, preX, preY, Grid_data, Tx_Grids)

- X, Y: represents the grid that this node is currently in.
- preX, preY: represents the last grid that this node was in
- Grid_Data: stores the list of data related to a particular grid
- Relevant Grids – Tx_Grids: a list of grids that data relates to these grids will be transmitted

Grid_Data(GX, GY, data_list)

- GX, GY: the Grid's location
- data_list: the prioritized list of data which are related to grid (GX, GY).

3.5.3 What to Transmit

The node transmits the data in Grid_Data only if this Grid_Data's Grid location exists in Tx_Grids by comparing each Grid's X of Tx_Grids with each Grid_Data's GX and each Grid's Y of Tx_Grids with each Grid_Data's GY, if both of them are the same, then transmit the data in this Grid_data in the prioritized order which will be introduced in Section 3.8. If there is more than one Grid in Tx_Grids, then the data relates to Grid with higher position in Tx_Grids has the higher transmission priority (Figure 3.8).

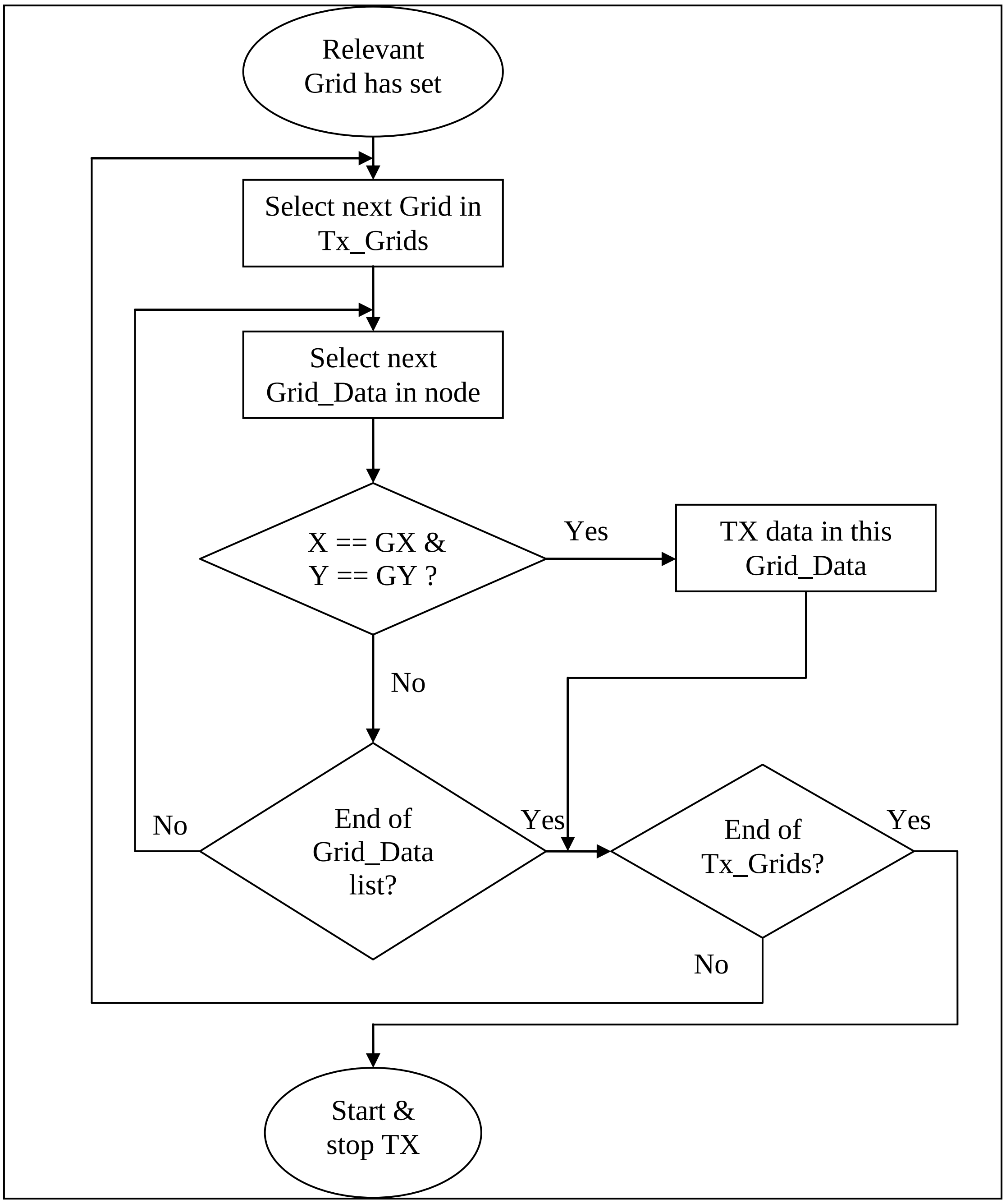


Figure 3.8 Flow chart of what to transmit for a node

3.6 Benefits of When to Transmit and What to Transmit Policies

With these proposed when to transmit and what to transmit policies, it is not necessary for each node to transmit all its carrying data at all the time when it is travelling within the network area, and hence only relevant data will be transmitted. The data could hop back to its area of origin by storing in the memory of the nodes which are already inside or moving towards the area which is the same as the area of origin of the data. In a sensor network with limited bandwidth, this would results the higher data availability by transmitting fewer data.

Data stored in the memory of each node are organized into groups depending on their collecting locations. Each group of the data will be transmitted when its area of origin is the previous leaving Grid of its carrier or its area of origin is the previous leaving Grid of the previous leaving Grid of its carrier. The group of data related to previous leaving Grid has the higher transmission priority. This is because the number of neighbouring nodes which intend to be in the previous leaving Grid is higher. In addition, the transmission of data which are related to other Grids is the responsibility of the other nodes. Thus, the nodes have different transmission tasks depending on their moving path, and this balances the work load of each individual node.

3.7 A Scenario that Hovering Information Works Better

As introduced in Section 2.3.1, a different approach has been proposed in Hovering Information project [43]. In this approach, each data is associated with an area of interest, a safe area and a risk area. All these areas are rounded area which centres at the sampling location of the data. A piece of data will be transmitted by its carrying node when its carrier is travelling in the region between its risk area and safe area. Thus, the data will be transmitted depending on its sampling location and the current location of its carry node. With the “when to transmit” and the “what to transmit” policies defined in this thesis, the node transmits a data together with other data which are related to the same Grid instead of transmitting them individually. This enables the capability to prioritize these data, in order to ensure the retained data represents more information. However, in the scenario illustrated in Figure 3.9, Hovering Information would work better than the defined “when to transmit” and the “what to transmit” policies. In this scenario, a node (N) is carrying some data including the data (D) whose sampling location is near the boundary of Grid (G3). In Hovering Information, the node does not need to transmit this data when it travels from G3 to G1 via G4 and G2, because it has not moved out of the safe area of the data. Thus, this data will still be available in its relevant area. However, the node, which complying with the “when to transmit” and the “what to transmit” policies, will start transmitting this data to other nodes when it is leaving G3. In this case, this data would be unavailable to its relevant Grid if there are no nodes are travelling to same Grid as its relevant Grid or the node N did not have sufficient time to transmit this data to other nodes. In addition, it also requires extra energy for transmitting this data.

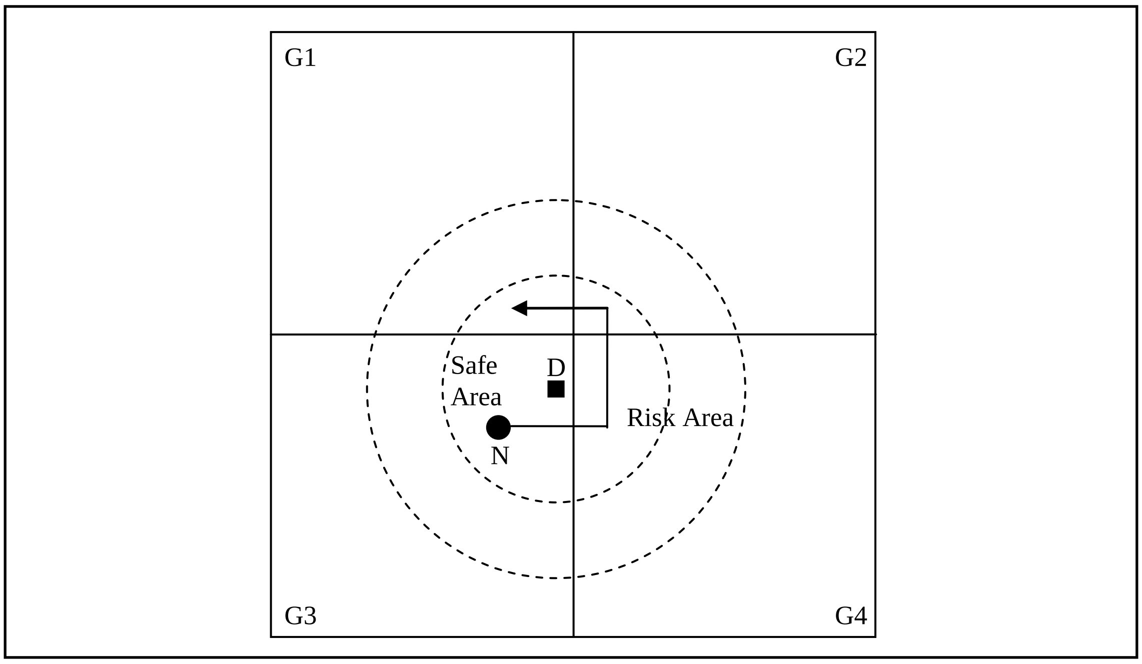


Figure 3.9 Scenario that Hovering Information works better

3.8 Data Prioritization

3.8.1 Overview

As mentioned in what to transmit policy (Section 3.5), the data stored in a single node are organized into groups based on their truncated sampling locations, and they will be transmitted in a prioritized order when transmission starts. The data prioritization of a Grid_Data (the term "Grid_Data" has been defined in Section 3.5.1) is not very important if all of the related data can be transmitted to other nodes between the period of start transmission and stop transmission of data of a particular Grid. However, in a mobile wireless sensor network where the network bandwidth is limited and the network nodes are not fixed in their original positions, data can only be transmitted when the receiver is within communication range of the sender. Thus, it is unlikely that all the data of a Grid_Data of a single node can be transmitted within this possible transmission time. This phenomenon can be affected by three factors: network bandwidth, size of data, and nodes' moving speed. In this case, data prioritization is significant to ensure that the most important data will be transmitted in higher priority order when data transmission started.

If two pieces of data were taken from the close locations and close sampling times, then these two pieces of data are likely to represent the same information. Once one of these data has been transmitted, it is better to set another to lower priority, in order to transmit the relevant data which represents different information as much as possible within the limited transmission time period. To achieve this purpose, there are two different methods which can be performed:

1) "Temporal prioritization": to ensure that the retained data related to a single grid are spread over the sampling time period
2) "Spatial prioritization": to ensure the retained data related to a single grid are spread over the space of its relevant grid

This thesis focuses on designing a temporal prioritization policy. The spatial prioritization would use the similar methods to the temporal prioritization, but is left for the future work (see Section 6.2.1).

The data prioritization will be performed when a node received a piece of data from another node. The received data will be inserted to the specific position of the appropriate Grid_Data, where this position is specified by different data prioritization policies. The data has higher transmission priority if it is at the higher position in the Grid_Data (Figure 3.10).

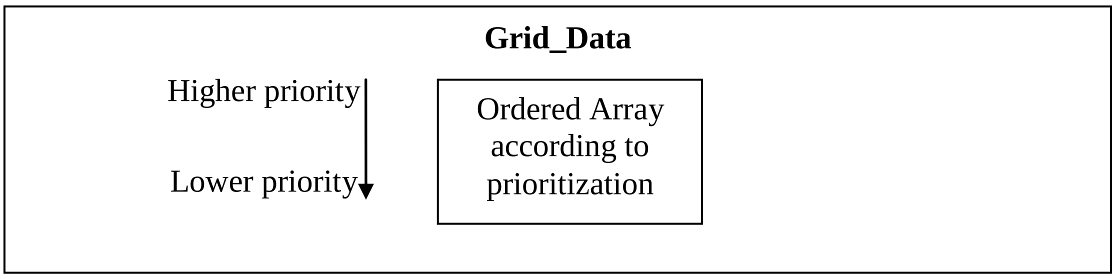


Figure 3.10 The priority of data in a Grid_Data of a network node

This section introduces three different data prioritization policies: “random prioritization”, “random temporal granularity prioritization” and “adaptive temporal granularity prioritization”. It is expected that the temporal quality of the retained data of random temporal granularity prioritization would be better than the random prioritization, and the adaptive temporal granularity prioritization would achieve the best temporal data quality. These different prioritization policies will be evaluated through the simulation which will be described in Chapter 5.

3.8.2 Random Prioritization

3.8.2.1 Defined Terms

Data – D(Dx, Dy, DX, DY, Dt)

- Dx, Dy: represents the sampling location of this data
- X, Y: represents the grid that this data belongs to.
- Dt: data's sampling time

3.8.2.2 Random Prioritization

When a node receives a piece of data, it selects the appropriate Grid_Data for this data and checks whether there is a duplicate copy in this Grid_Data by comparing the data's sampling location (Dx, Dy) and sampling time (Dt) with the existing data. If none of the existing data has the same sampling location and sampling time, then there is no duplicate copy. The data will be then inserted to a random position in the appropriate Grid_Data (Figure 3.11). Thus, the Grid_Data of different nodes with the same identification (GX and GY) are likely to maintain a list of data with different prioritized order, so that when transmission of data starts, the data transmission order for different nodes will be different.

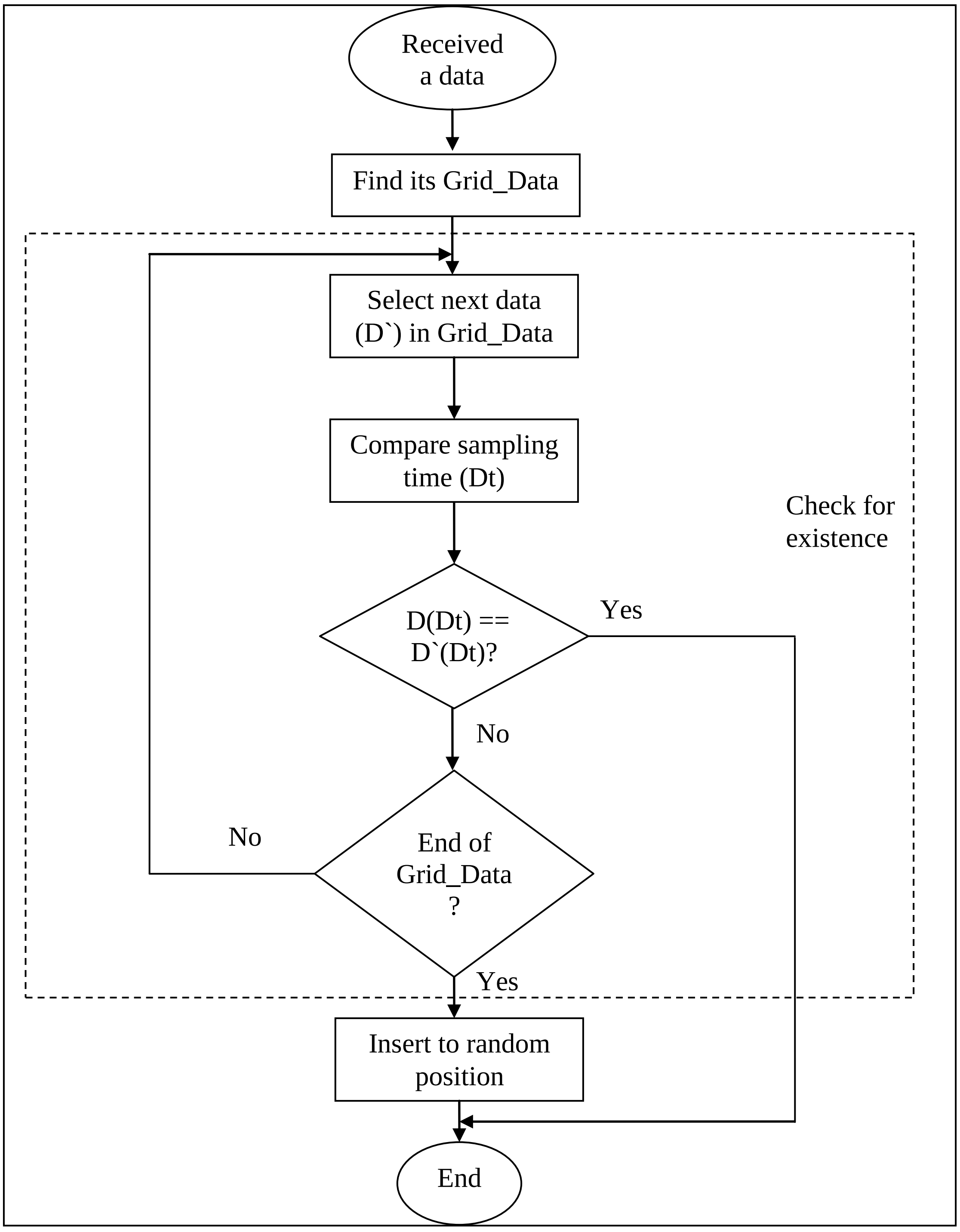


Figure 3.11 Flow chart of random prioritization

3.8.3 Random Temporal Granularity Prioritization

3.8.3.1 Defined Terms

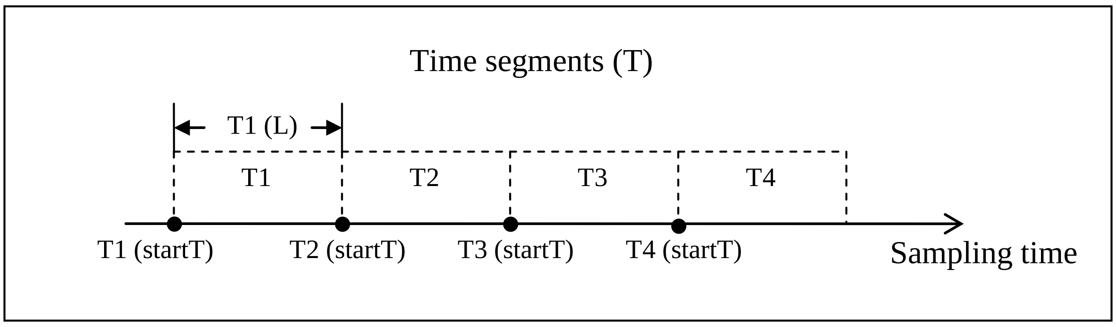


Figure 3.12 The time segments of random temporal granularity prioritization

Time segment – T(startT, L): The sampling time period of data will be divided into smaller identical time segments. The time intervals of these time segments are the same, and the length of this time interval between any two segments is depending on the length of sampling time period and the total number of time segments. Figure 3.12 illustrates an example of time segments.

- startT: the starting time of this time segment. Different time segments have the different starting time.
- Time segment interval – L: the time interval of each time segment.

Data – D(DX, DY, Dt, DT)

- DX, DY: represents the grid that this data belongs to.
- Dt: data’s sampling time
- DT: the time segment that this data belongs to. To compute which time segment that this data is belonging, the data’s sampling time will be compared with each time segment to check whether it is between the start time and end time of any

time segment. Thus, if Dt >= startT and Dt < startT + L, where the startT and L are the start time and the time segment interval of a time segment, then this data belongs to such a time segment.

3.8.3.2 Random Temporal Granularity Prioritization

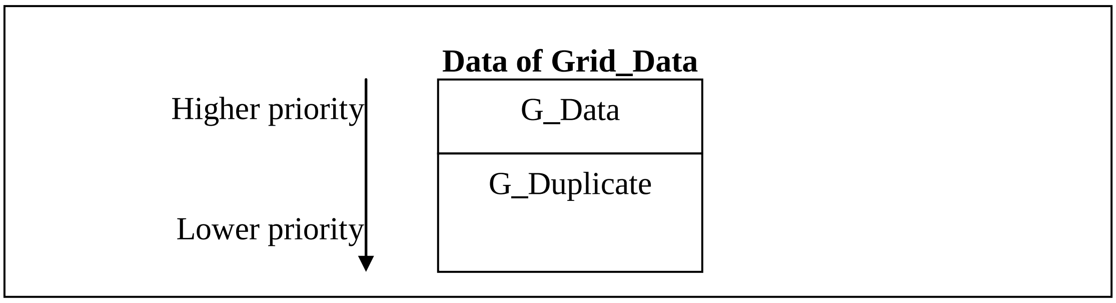


Figure 3.13 Transmission priority of data in Random Temporal Granularity Prioritization

The random temporal granularity prioritization prioritizes the data of the Grid_Data into 2 groups depending on the time segments that the data belongs to: G_Data and G_Duplicate. G_Data stores the data which belong to the different time segments, and G_Duplicate stores the rest of data whose belonging time segment is the same as any of the data in G_Data. The data stored in G_Data has a higher transmission priority than the data stored in G_Duplicate (Figure 3.13). The data with higher position in G_data has the higher transmitting priority and the same rule applies for G_duplicate.

Each node builds its own list of time segments after it has been deployed. The number of time segments in a list is varied for different nodes. This number has a value 2^n, where n is an integer between 1 and $\log_2$ (total number of initial data located in a Grid) - 1. Thus, the minimum value of this number is 2 and the maximum value of this number is the quotient of dividing the total number of initial data located in a Grid by 2. Each node then chooses a random integer of n within the range of n. Having the various total numbers of time segments, nodes may have different lists of time segments with different time segment intervals.

When a node receives data from another node, it computes the data's belonging time segment by checking whether the sampling time of the data is within the time interval of each time segment. It then compares this belonging time segment with the time segment of each data in G_data. If any of them has the same time segment (i.e. time segment with the same starting time), then it inserts the received data to a random position of G_duplicate, otherwise it inserts it to a random position of G_data. Figure 3.14 illustrates the flow chart of this procedure.

Since the data is storing in random positions of both G_Data and G_Duplicate, the transmission priority of these data are also random. By means of this, different nodes may have different transmission priorities. This would result more data being transmitted by different nodes which have the same previous leaving Grid. In addition, the data storing in G_Data were collected from different segments of the sampling time period and these data have the higher transmission priority. Thus, this would lead to more data representing different information being transmitted.

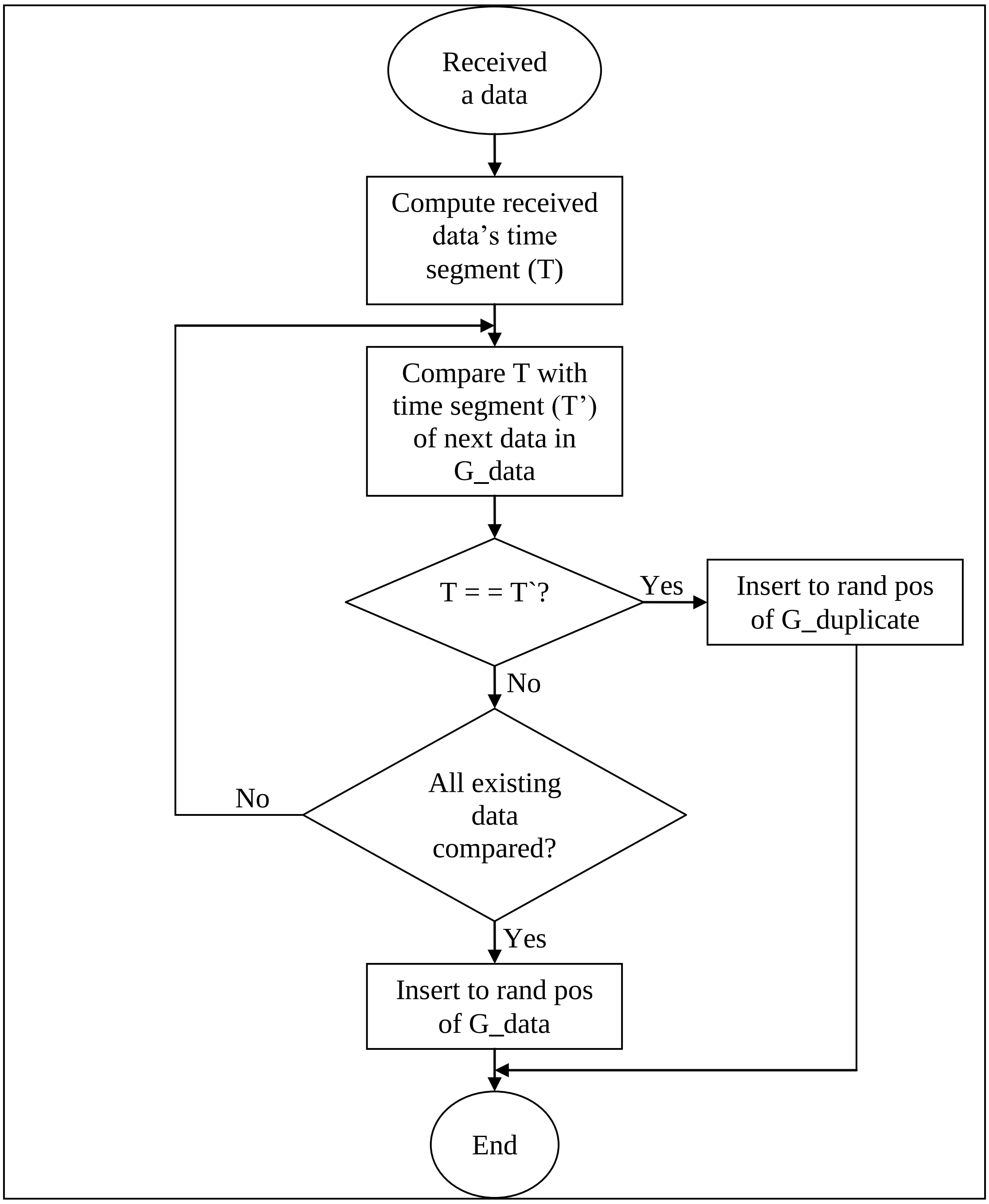

Received a data
Compute received data's time segment (T)
Compare T with time segment (T') of next data in G_data
T = = T`?
Yes
Insert to rand pos of G_duplicate
No
All existing data compared?
No
Yes
Insert to rand pos of G_data
End

Figure 3.14 Flow chart of Random Temporal Granularity Prioritization

3.8.3.3 Example of Random Temporal Granularity Prioritization

Figure 3.15 shows an example of the random temporal granularity prioritization. Considering a node holding 6 data related to the same grid, and these data were taken from 3 time segments of the sampling time period: 2 data from time segment T1, 3 data from time segment T2, and 1 data from time segment T4. Based on this prioritization policy, only one piece of data collected from the same time segment will be stored in G_Data and others will be treated as duplicate data and they will be stored in G_Duplicate. Therefore, G_Data stores 3 data which are taken from time segments T1, T2 and T4, G_Duplicate stores 3 data which are taken from time segments T1 and T2.

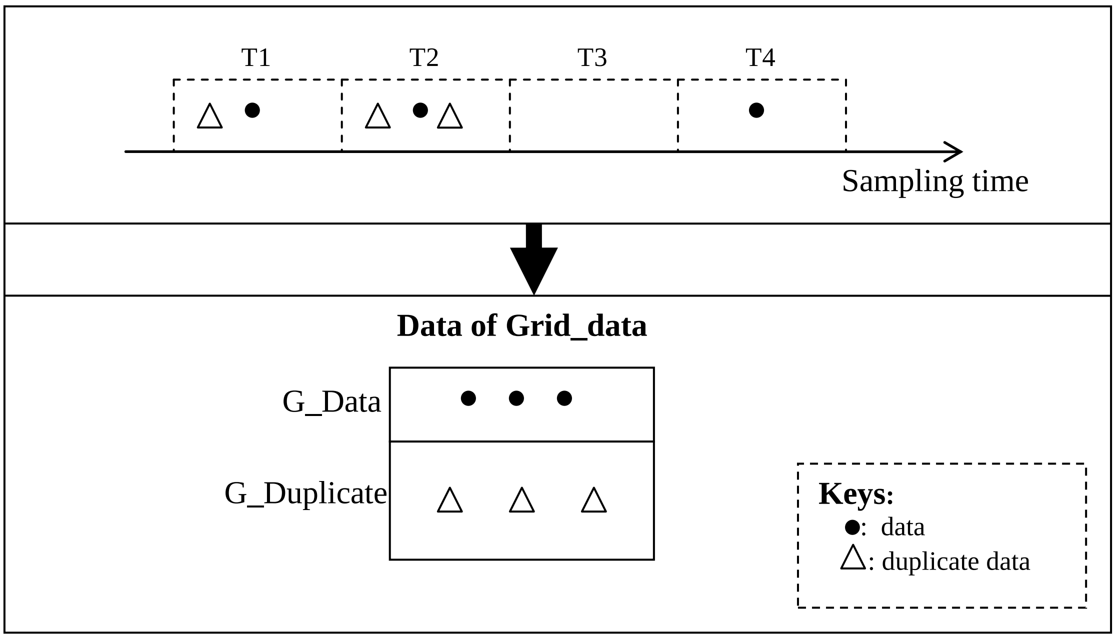


Figure 3.15 Example of Random Temporal Granularity Prioritization

3.8.4 Adaptive Temporal Granularity Prioritization

3.8.4.1 Defined Terms

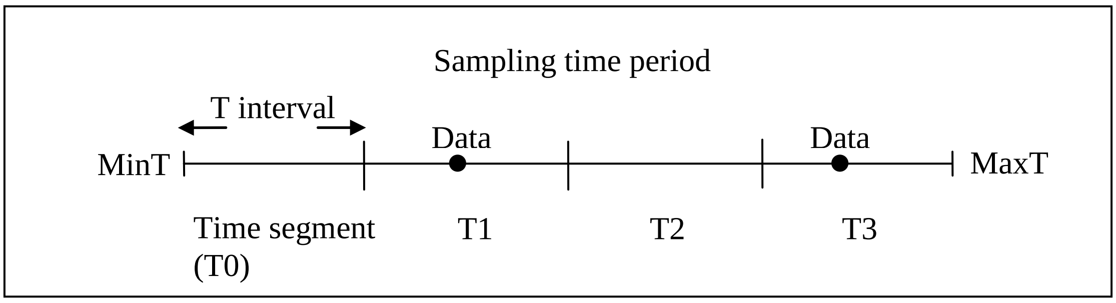


Figure 3.16 Defined terms of Adaptive Temporal Granularity prioritization

Sampling time period: is a period of time that all the initial data was being taken from a particular grid (see Figure 3.16). The sampling time period has a fixed length, as we assume that the nodes will not sense after network deployment, thus there is no more data will be created.

- MinT: The earliest data sampling time among all data taken from a grid.
- MaxT: The latest data sampling time among all data taken from a grid.

Data – D(DX, DY, TIndex, Dt)

- DX, DY: represents the grid that this data belongs to.
- TIndex: the index of the time segment that this data's sampling time belongs to.
- Dt: data's sampling time

Time segment – T(startT, L, TIndex): The sampling time period will be divided into smaller identical time segments where this number is equal to the number of initial data of a particular grid (i.e. 2^n, where n is a positive integer). (For example, in Figure 3.16, the sampling time period is divided into 4 time segments T0, T1, T2 and T3.) The time intervals of these time segments are the same.

- startT: the starting time of this time segment. Different time segments have the different starting time.
- T interval – L: the time interval of a time segment. This time interval is calculated by dividing the difference between MaxT and MinT by number of time segments: L = (MaxT - MinT) / 2^n.
- Time segment index – TIndex: each time segment has a unique index, and this index will be assigned when deploying the node: the time segment starting from MinT has index 0, and the one with later starting time has index 1 and so on.

Grid_data(GX, GY, TIndex_Data): (see Figure 3.17)

- GX, GY: is the grid's location
- TIndex_Data (TIndex, Data): is an array of ordered time segment indices associating with the corresponding data where its position in the array is the same as its calculated time segment index (see Figure 3.17).

Grid_Data (GX, GY)

TIndex_Data	
TIndex	Data

Figure 3.17 TIndex_Data in Grid_Data for adaptive temporal granularity prioritization

3.8.4.2 Adaptive Temporal Granularity Prioritization

Adaptive temporal granularity prioritization has 2 major procedures: 1) construct the list of time segment indices (TIndex) in TIndex_Data by adapting the temporal granularity and ordering the indices of time segments, and 2) prioritize the data by storing the data to an appropriate position in the TIndex_Data of Grid_Data based on the data's calculated time segment index.

When a network node has been deployed, procedure 1 and 2 will be carried out: the node will construct the time segment indices list for each Grid_data and then reorder the existing data. Once the time segment indices list of a node's Grid_Data has been built, the order of these indices will be remained constantly. When a node received a new piece of data, only procedure 2 will be carried out: the node inserts the received data to the appropriate position in the TIndex_Data of a particular Grid_data.

<u>Construct Time Segment Indices List</u>

This procedure has been designed based on the Adaptive Tree Walk Protocol [47]. The list of time segments indices is initially empty. To construct the ordered time segment indices list, a loop of steps of adapting the temporal granularity of sampling time period will then be carried out. In this loop, the sampling time period will be divided into 2^m (where m is a positive integer with initial value 1, and it will be increased by 1 for each run of the loop) groups of time segments until the current number of groups is equal to number of time segments (i.e. $2^m = 2^n$). For each run of the loop, the smallest index of each group of time segments will be selected to compare with the existing indices in TIndex list of TIndex_Data; if none of the existing indices has the same value, and then insert this smallest index to the end of TIndex list (see Figure 3.18).

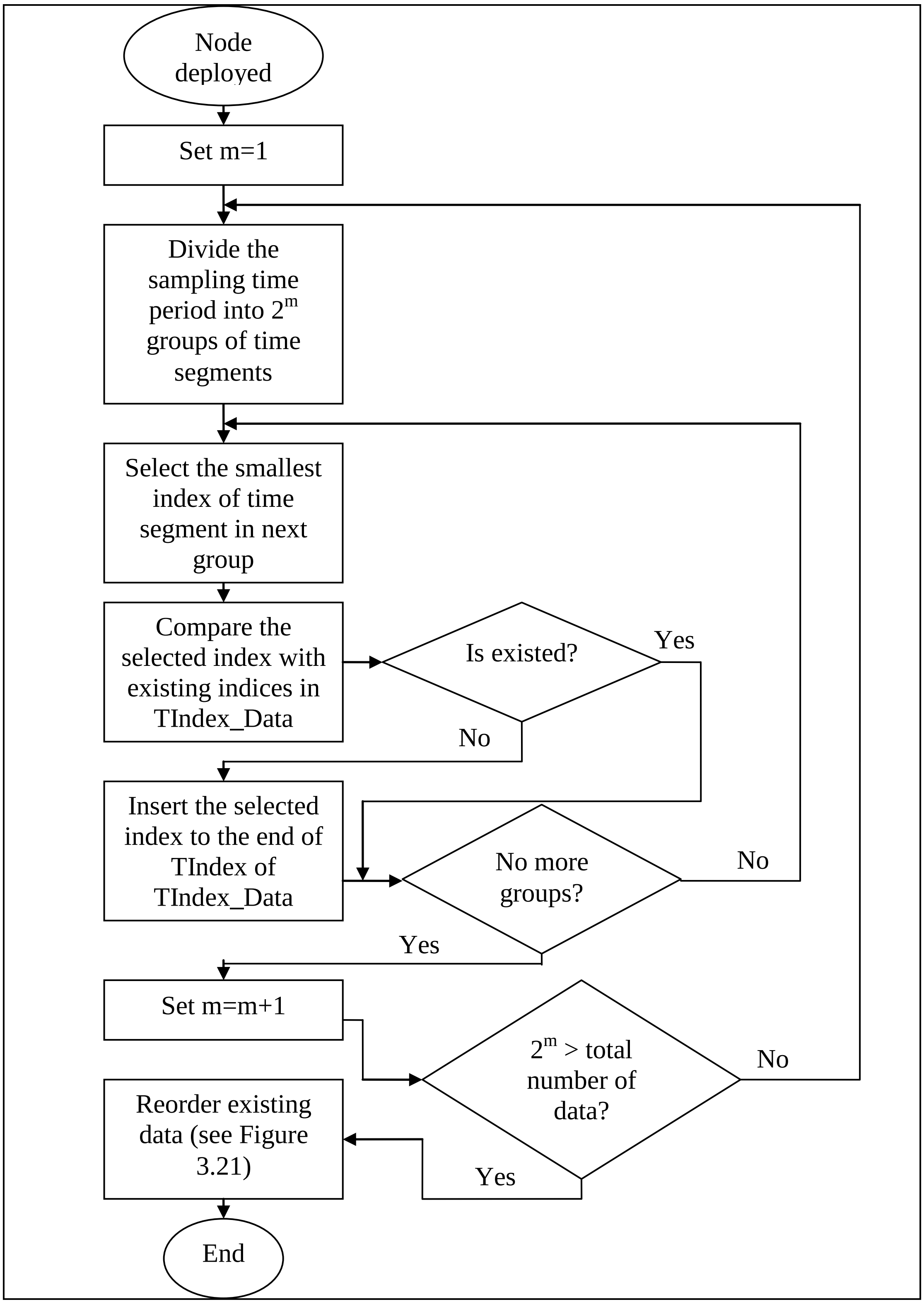

Node deployed
Set m=1
Divide the sampling time period into 2^m groups of time segments
Select the smallest index of time segment in next group
Compare the selected index with existing indices in TIndex_Data
Is existed?
Yes
No
Insert the selected index to the end of TIndex of TIndex_Data
No more groups?
No
Yes
Set m=m+1
2^m > total number of data?
No
Yes
Reorder existing data (see Figure 3.21)
End

Figure 3.18 Flow chart of constructing time segment indices in Adaptive Temporal Granularity prioritization

Figure 3.19 shows an example of constructing time segment indices for a Grid_Data with 8 initial data related to this grid. In the first division of the sampling time period (labelled ½), there are 2 groups of time segments, and indices 0 and 4 will be inserted to the end of TIndex list. In second division, indices 0, 2, 4, and 6 are the smallest time segment indices, but indices 0 and 4 are existed in the TIndex list, so that only 2 and 6 will be inserted. For the third division, the rest of indices which are not existed in TIndex list will be inserted.

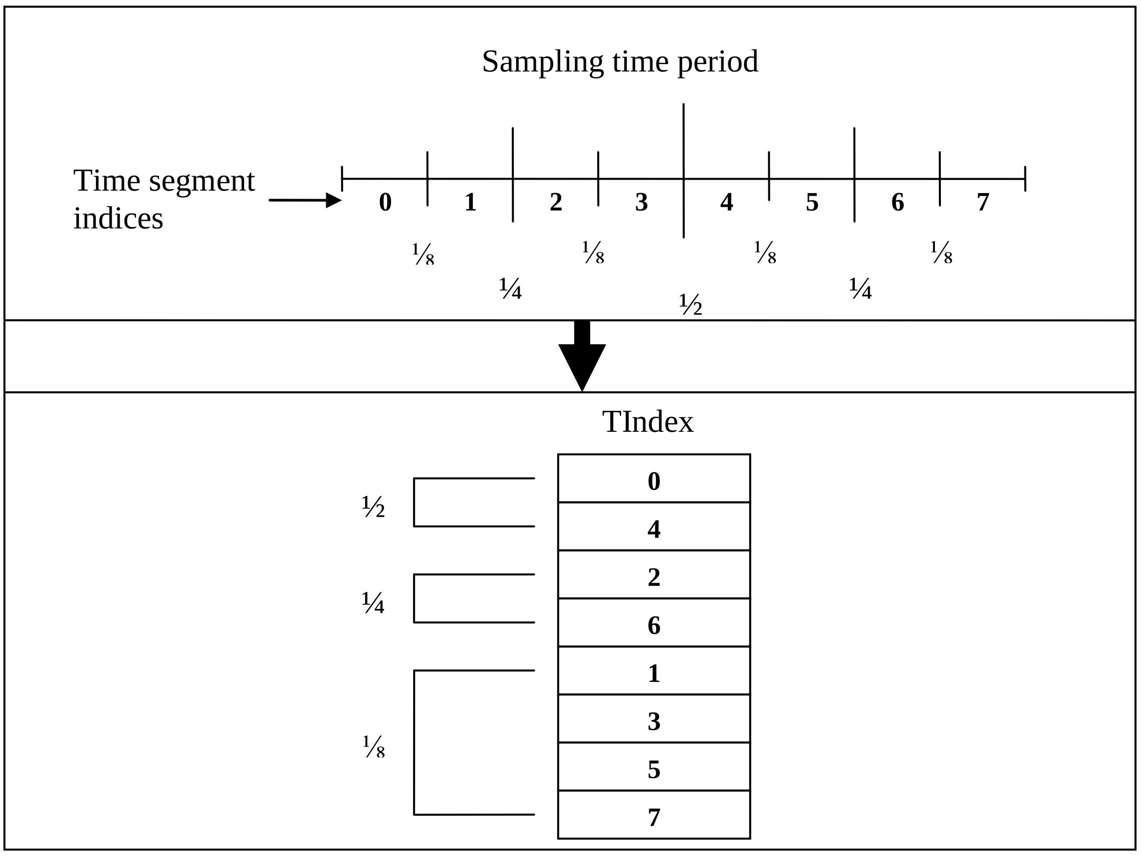


Figure 3.19 Example of constructing time segment indices

Alternatively, there is another method for achieving the similar result by representing the indices in binary number system (see Figure 3.20 as an example):

1) Represent the indices in binary

2) Order these binary numbers in ascending order

3) Reverse the bits of each binary number

4) Insert these new binary numbers into TIndex with their current sequence

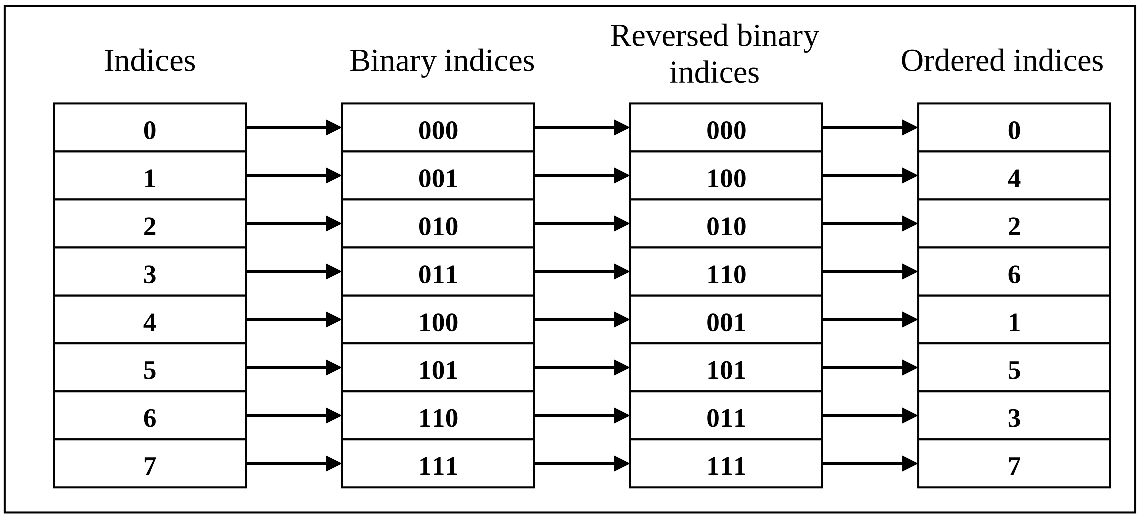


Figure 3.20 Example of constructing time segment indices by reversing bits

Data Prioritization

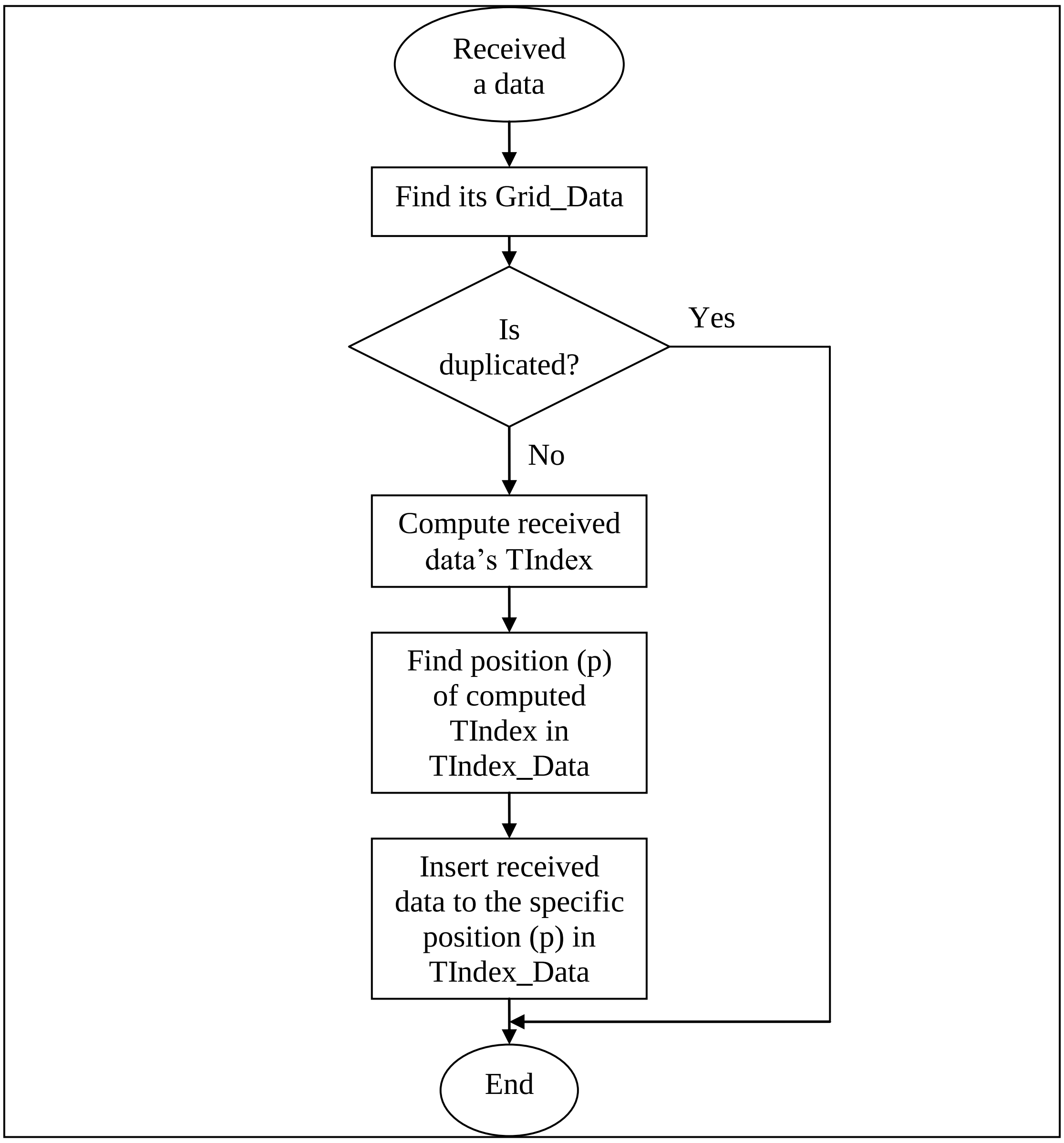

Received a data
Find its Grid_Data
Is duplicated?
Yes
No
Compute received data's TIndex
Find position (p) of computed TIndex in TIndex_Data
Insert received data to the specific position (p) in TIndex_Data
End

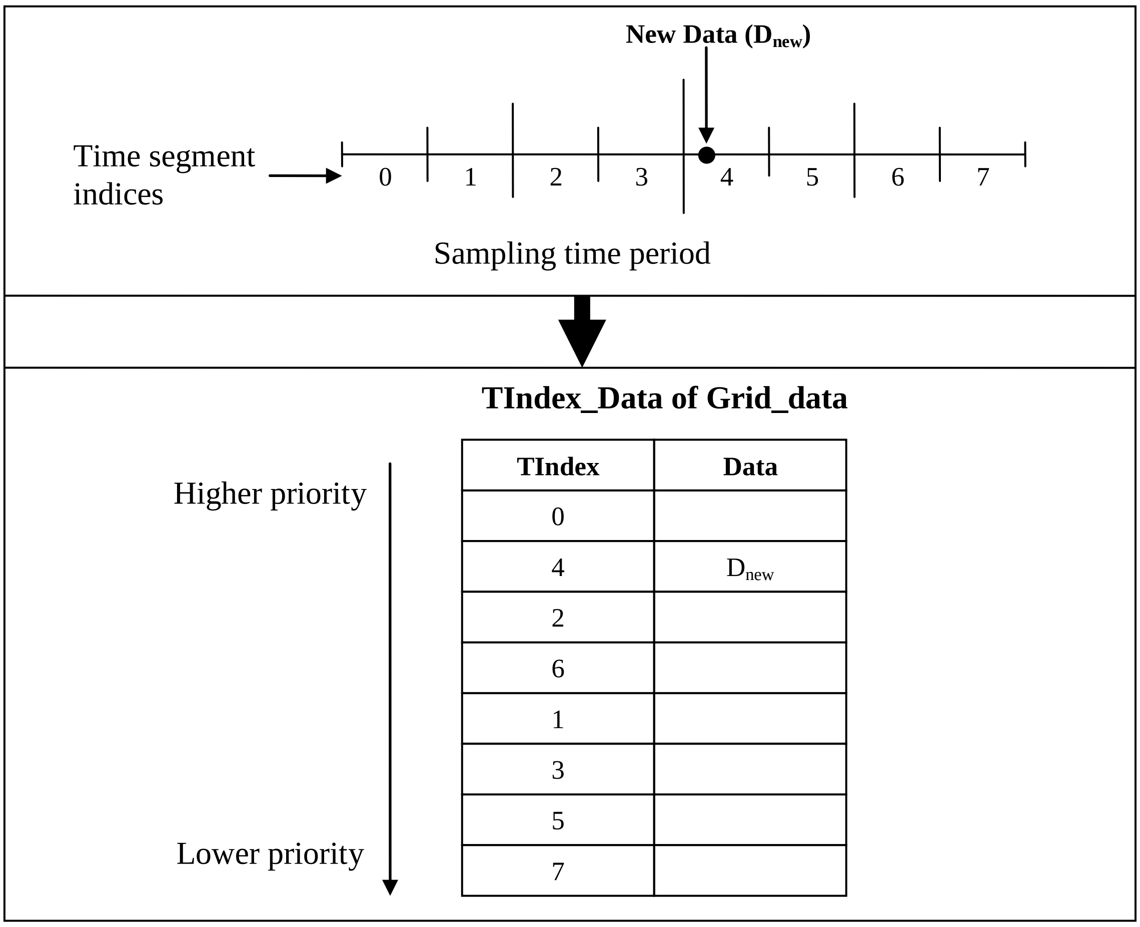
Figure 3.21 Flow chart of reordering data in Adaptive Temporal Granularity prioritization

This data prioritizing procedure (see Figure 3.21) can be triggered in 2 possible situations: 1) when receiving a new data and 2) after the time segment indices list has been built. When a node received a new data (D_{new}), the following steps will be carried out to insert this data into the TIndex_Data of Grid_data at appropriate index position.

When the time segment indices list has been built, all the existing data in the node have to be reordered. The process of reordering the existing data can be thought to be receiving several new pieces of data, and these received data are the existing data. Thus, the following steps will be carried out for each of the existing data. (Figure 3.22 shows an example of inserting a received data to the appropriate position in TIndex_Data after the time segment indices list has been built.)

1) Calculate the new data's index by dividing the difference between the new data's sampling time and MinT by time segment interval (D_{new} (TIndex) = (D_{new} (Dt) - MinT) / L).

2) Insert the new data to the TIndex_Data array at the same position as its calculated TIndex.



TIndex	Data
0	
4	D_{new}
2	
6	
1	
3	
5	
7	

Figure 3.22 Example of data prioritization in Adaptive Temporal Granularity prioritization

3.8.4.3 Complexity of Reordering

In the “Adaptive temporal granularity prioritization”, each data is associated with a computed “time segment index”. This index is computed based on the data’s sampling time, the total number of data within the same Grid which are carrying by the same node, and the sampling time period of these data. To prioritize the data of a node, the time segment indices will be reordered.

As discussed in this Section (Figure 3.18), this “Adaptive policy” reorders the time segment indices by selecting the appropriate time segment index and compare this index with the existing indices which are currently in the list of prioritized indices (TIndex) to check for existence. If no duplicate index has been found in this list, then this index will be inserted to the end of the TIndex. The number of comparisons for each selected index is equal to the current number of indices in the TIndex. This number will be increment by one when a selected index has been inserted. For these selected indices, some indices may already exist in the TIndex. Thus, the total number of comparisons is equal to the sum of the total number of comparisons for all non-existing selected indices and the total number of comparisons for all selected indices in which each has at least one duplicate index in TIndex. The number of comparisons of non-existing indices is equal to 0+1+…+ (n-1) where n is the total number of time segment indices. The number of comparisons of duplicate selected indices is equal to 2+3+…+ (n-1). This because the first 2 selected indices will not have duplicate copies in the TIndex based on this defined policy, the index may only has duplicate copy from the third selected index, and this index will compare with two existing indices in the TIndex. Thus, this total number of comparisons starts adding from 2. Therefore, the number of comparisons for reordering the time segment indices is equal to $(n-1)*n/2 + ((n-1) +2)*(n-2)/2 = n^2 – n – 1$. The complexity for reordering time segment indices is O (n^2).

As shown in Figure 3.21, when a node receives a new data from another node, it computes the time segment index of this data and inserts this data to the appropriate position by comparing its computed index with existing indices in the TIndex. For the worst case, the number of comparisons of this procedure is equal to the total number of indices in TIndex (n). Thus, its complexity is O (n).

The complexity of reordering data in the “Adaptive temporal granularity policy” is O (n^2) + O (n) = O (n^2).

Chapter 4

Simulation and Evaluation Methodology

4.1 Introduction

In Chapter 3, the defined policies of the Data Hovering algorithm and the baseline approach have been introduced. The Hovering Information broadcasting algorithm, which has been introduced in [43], will be simulated and evaluated to contrast with the defined policies. This algorithm and the defined Data Hovering algorithm will then be implemented in a simulator. This chapter introduces the details of methodologies of simulation and evaluation after the system implementation.

4.2 Simulation Methodology

4.2.1 Assumptions

The following assumptions for the simulation have been made in order to enable the capability for evaluation:

- Unlimited memory: All nodes have unlimited memory which is able to store numerous data. The defined Data Hovering algorithm in this thesis will address the constraints including limited energy and limited bandwidth which have been described in Section 1.2.3. Thus, addressing the issue of limited memory will be left for the future work in Section 6.2.2.

- No sensing: The network nodes will not take samples when moving in the network, but each node initially carries some data that was taken from this network area. Since the Data Hovering algorithm will be evaluated by measuring the ratio of number of retained data to the total number of data which are related to their area of origin, adding more data will not affect this ratio but increase the complexity for measuring. This assumption ensures that the total number of data to transmit remains constant.
- The initial data related to each grid is evenly spread over the sampling time period, and they are evenly spread over their related grid
- The total number of initial data related to each grid is equal to 2^n, where n is a positive integer.
- Total number of nodes in the network area remains constant: This is because the number of network nodes has a significant impact on data availability. Without applying the Data Hovering algorithm, the data availability is directly proportional to the total number of nodes, because more nodes will be located in each area of origin of the data.

4.2.2 Simulation Environment

The Data Hovering algorithm has been implemented and will be simulated in OMNeT++ [48] network simulator (distribution 3.3) with Castalia [49] (version 2.0) extension to support Wireless Sensor Networks. The simulation experiments will be run on a Pentium 2.8 GHz processor under Fedora 10.

4.2.3 Network and Grids

The size of network area for simulation is 400m x 400m (as shown in Figure 4.1). To form a square network area, the total number of Grids must be n^2, where n is a positive integer. As described in Section 3.4.2, a node will transmit its data related to the Grids which currently exist in its Tx_Grids. The contents of the Tx_Grids will be changed based on the current location of the node. Thus, this requires at least 9 Grids, when the node is moving in a straight line in the network area (in the mobility model using in this simulation, each node will be moving in a straight line, see Section 4.2.4). Therefore, in this simulation, the network area further divides into 16 identical Grids, so the size of each Grid is 100m x 100m. Each grid will be assigned a unique location which is the coordinate of its top-left corner, so that it can be distinguished from other grids. For example, the Grid at the top-left corner of the network area has location (0, 0).

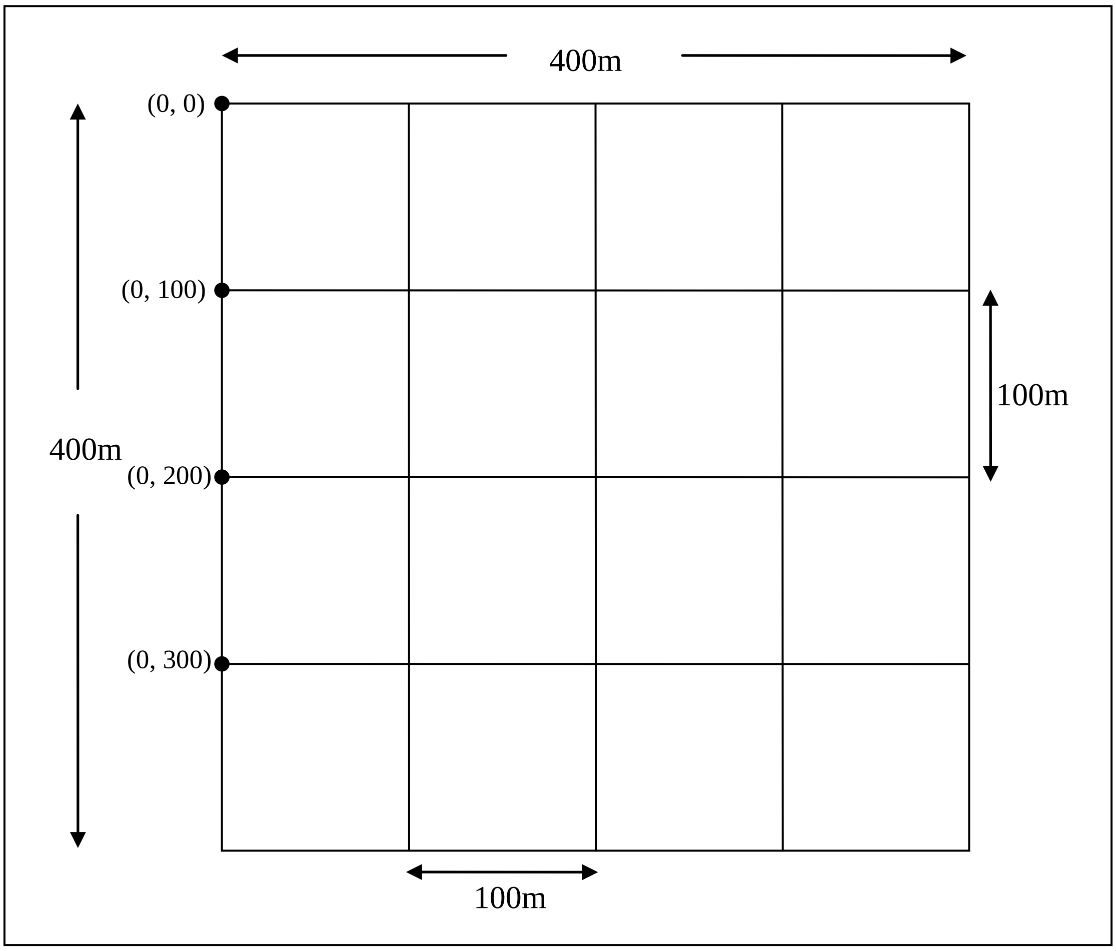


Figure 4.1 Simulation settings: network and grids

4.2.4 Mobility Model

A specific mobility model has been designed for the simulation of Data Hovering algorithm. In this mobility model, the network nodes move within the network area in which the details of this network area will be specified by user.

When a simulation starts, each individual node stays in its initial location for a certain time period, and this time period is called node **start moving delay**. Each node can have its own start moving delay, and this will be specified by user in simulation configuration file. (In this simulation, different nodes have different start moving delays. The setting of start moving delays of individual nodes and the reason of why different nodes have the different start moving delays will discuss in Section 4.2.6.) Once the start moving delay of a particular node expired, the node selects a random location on one of the boundaries of network area as destination, and then it moves towards the destination in a straight line with a constant speed. The moving speeds for all network nodes are the same. When a node reached its destination, it will stay at current destination for a while, which is called **node pause time**. After the pause time, the node will then start moving to another random location on one of the other network boundaries as next destination where this network boundary is not the same as the boundary of its current location (An example of this is illustrated in Figure 4.3, and Table 4.1 provides the behaviours of the node of this example). In a real network, the nodes can leave and enter the network area. This mobility model approximates this situation by maintaining constant number of nodes in the simulated area. In addition, nodes leaving the simulated area take data with them, and nodes entering the simulated area have no knowledge of this area. The concept of **Garbage Data** (see Section 4.2.8) will be used to approximate this situation by forcing nodes which reach a boundary to forget all their carried data before bouncing off the boundary.

In the real world, a network area is used to monitor an interesting region. Existing nodes inside this network area can move out of this area and other nodes can move into this area from other places without any knowledge of this area. This situation can be approximated by reflecting the nodes back to the network area when it is reaching the boundary of the network area, and along with the “Garbage Data” concept which will be discussed in Section 4.2.8. This mobility model also includes a “start moving delay” and a “node pause time”, because different nodes may start moving at different time and these nodes may pause at some locations for a certain time period. Typical application would be attaching the sensor nodes to the animals to track their movements. In addition, limited transmission time for mobile nodes is one of the major concerns in a mobile wireless sensor network, and it is also one of the requirements for simulating the defined “Data Hovering algorithm”. To ensure the transmission time is limited, the maximum bandwidth is necessary to be calculated. To easily calculate this bandwidth, the nodes complied with this mobility model are always moving straight lines in the network area with a constant speed, even though the defined algorithms will still work properly without these settings. Future mobility model would consist of more realistic features to enable the capability of various velocities with acceleration, and different node movements, thus the nodes are not only moving in straight lines.

The movement of each network node is a loop of selecting next destination, moving to the selected destination, and pause for a certain time, till the end of simulation or its power level is low. Figure 4.2 shows the flow chart of how mobility model works.

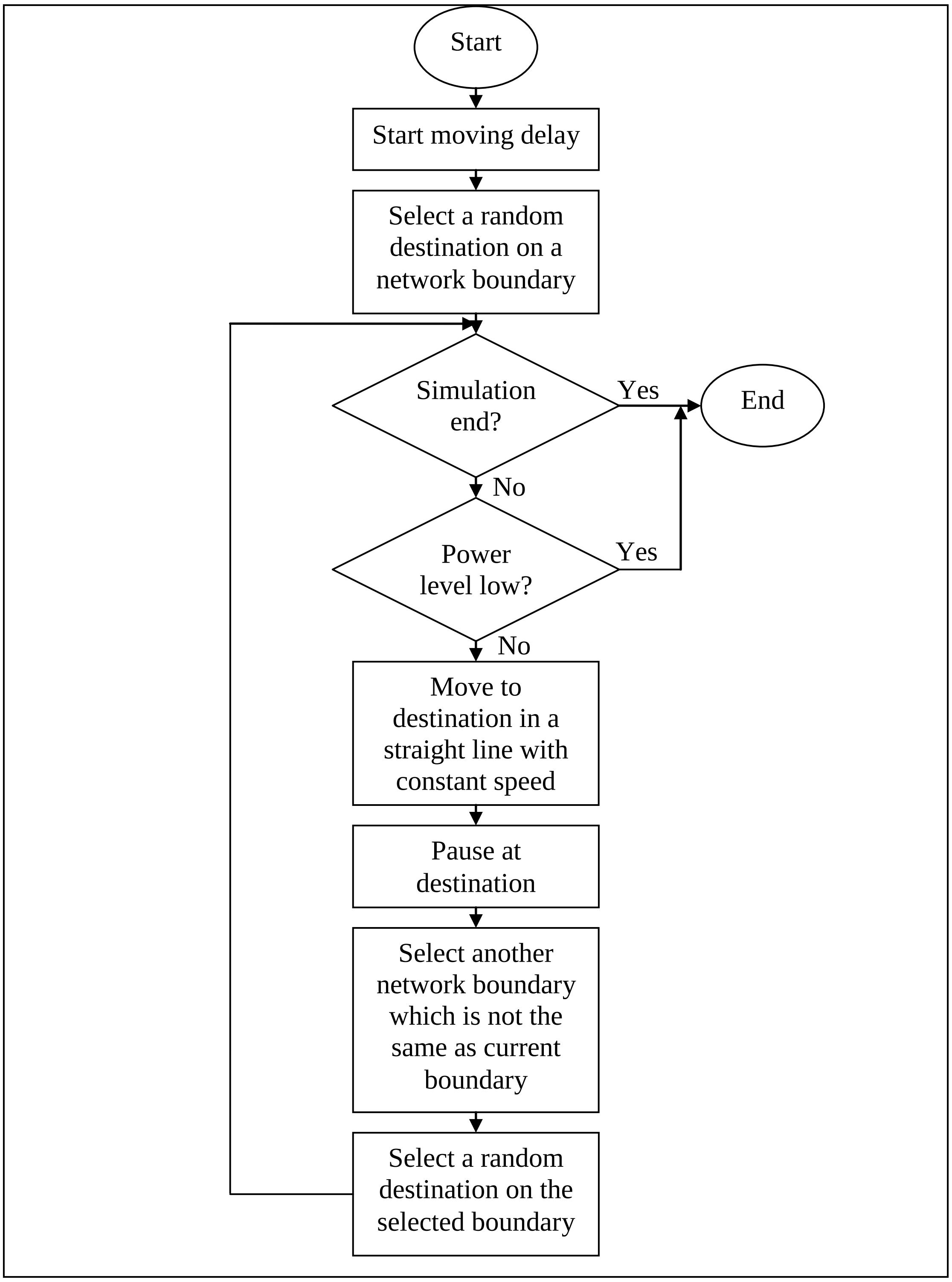

Start
Start moving delay
Select a random destination on a network boundary
Simulation end?
Yes
End
No
Power level low?
Yes
No
Move to destination in a straight line with constant speed
Pause at destination
Select another network boundary which is not the same as current boundary
Select a random destination on the selected boundary

Figure 4.2 Flow chart of designed mobility model

The movements of the nodes can be saved to a file, and this file can be loaded again for another simulation run if all the details of the network areas, number of nodes and the simulation time limits of both simulation runs are the same. This makes it possible that the moving path of each individual node in different simulation runs to be the same.

Figure 4.3 and Table 4.1 show an example of movement behaviour of a network node that complied with this mobility model. Figure 4.3 shows the movement path of this network node in a network area where A, B and C represent the locations of this node in its moving path, and b1, b2, b3 and b4 indicate the different network boundary of this network area. Table 4.1 shows the behaviour of this node at different locations.

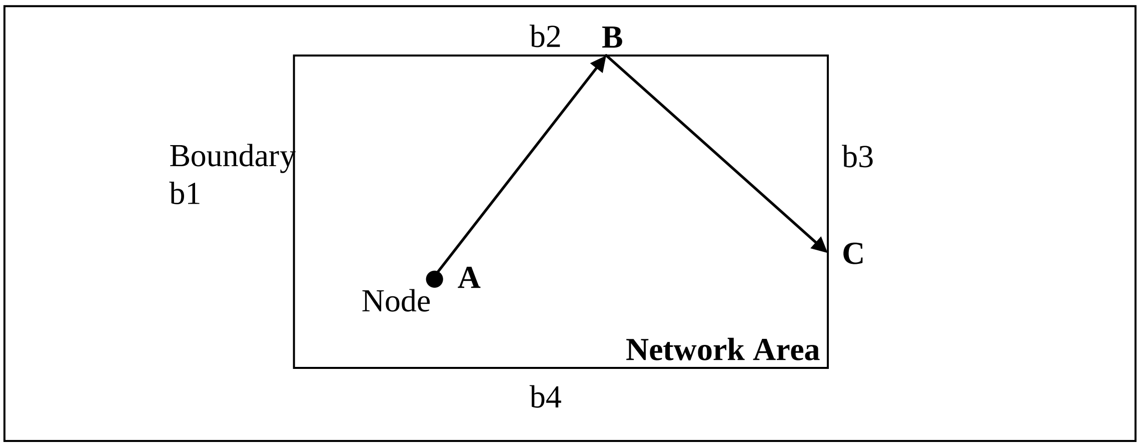


Figure 4.3 Example of movement behaviours of a network node complied with defined mobility model (1)

Current location	**Current boundary**	**Next destination**	**Next boundary**	**Description**
A	None	B	b2	Node starts at its initial location A, and stays till the moving starts delay expires. It then starts moving to a random location B on boundary b2.
B	b2	C	b3	Node reaches at destination B, and it waits for a specified pause time. It then selects random location C on another boundary (b3) which is not the same boundary as the boundary of its current location (b2), and it starts moving to next destination C.
C	b3	Random	b1, b2 or b4	Node reaches at destination C, waits for a pause time, and then moves to next destination

Table 4.1 Example of movement behaviours of a network node complied with defined mobility model (2)

4.2.5 Initial Data

In this simulation, there are 64 initial data which were taken from each Grid, so the total number of initial data of the network area with 16 Grids is 1024. Based on the assumptions of Data Hovering algorithm, nodes will not take samples during the simulation, and hence this total number of data represents the maximum number of non-duplicated data within the network. All these data were taken from a sampling time period 0-1024 seconds. The size of every initial data has set to 320 bits.

One of the aims of the Data Hovering algorithm is to improve the quality of the retained data, and this can be done by either temporal prioritization or spatial prioritization (as mentioned in Section 3.8.1). The prerequisite for achieving this aim is that the sampling times of initial data which were taken from the same grid must be evenly spread over the sampling time period and/or their sampling locations must be evenly spread over the space of their related grid. Figure 4.4 shows an example of how the sampling locations of initial data are evenly spread over their related grid and the sampling times are evenly spread over the sampling time period of 4 initial data of a single grid. The data's sampling locations and sampling time of this simulation which involving 64 data of each grid are initialized at the same rules as this example.

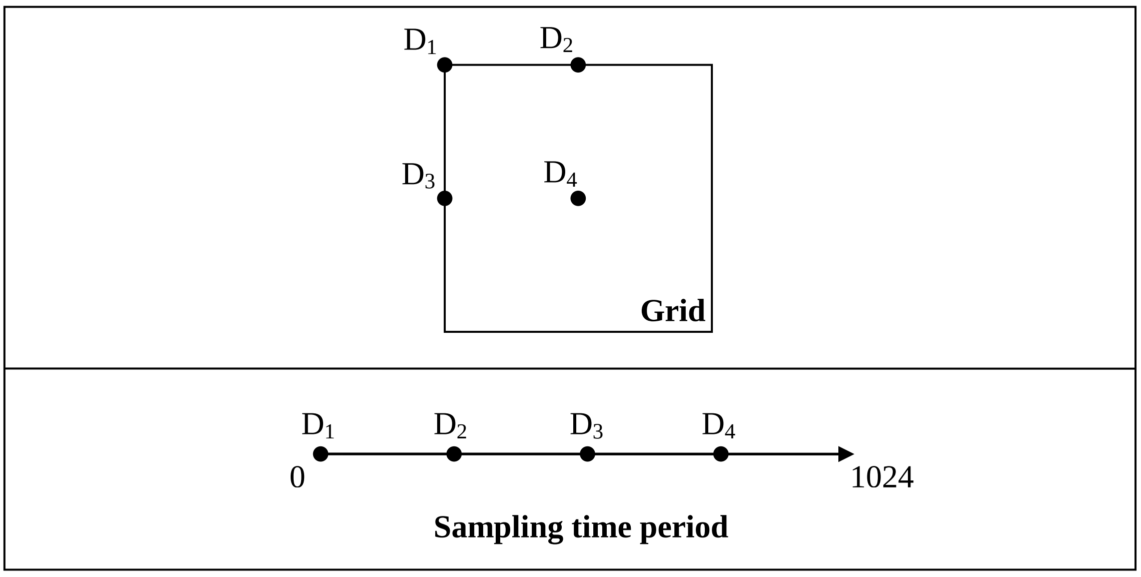


Figure 4.4 Simulation settings: initial data's sampling locations and sampling time (example of 4 data related to a single grid)

4.2.6 Nodes

There are 16 nodes initially deployed in each Grid, so that the total number of nodes within the network area involving 16 Grids is 256. The nodes which are deployed in the same Grid are initially located in centre of their initial Grid (as shown in Figure 4.5). Node i is denoted as N_i in the following presentation.

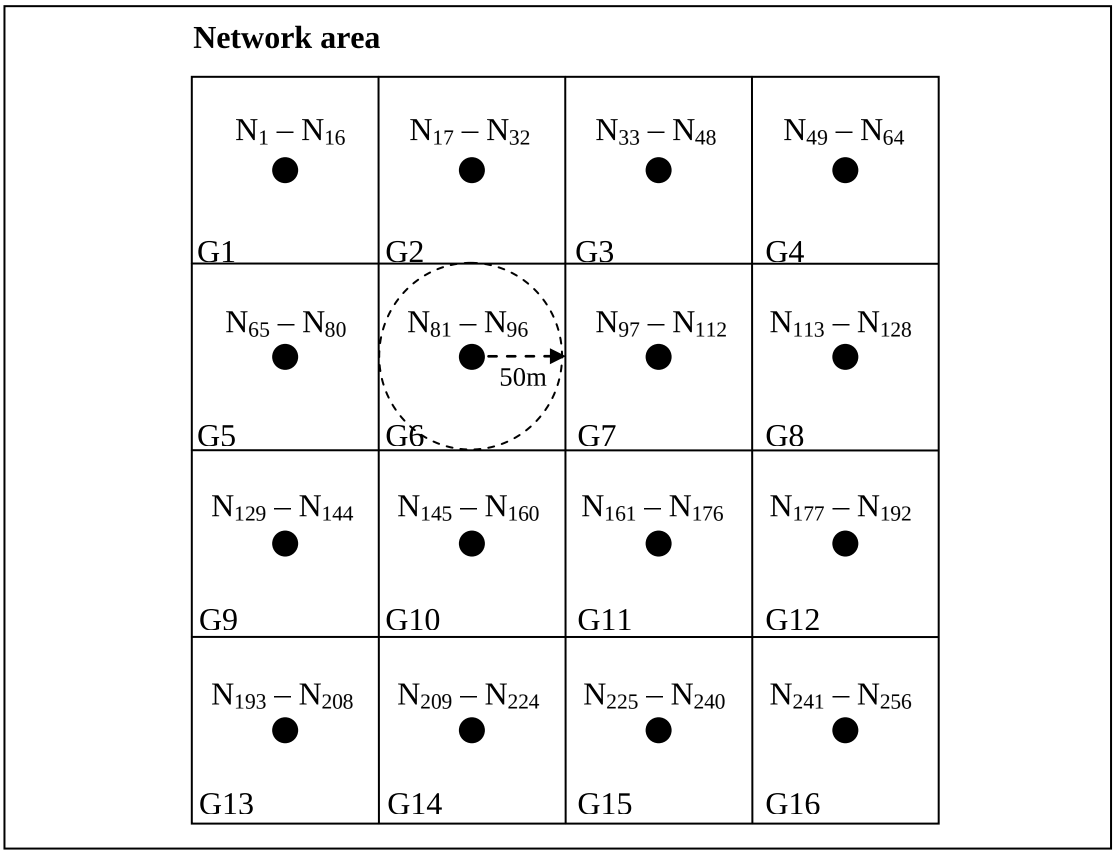


Figure 4.5 Simulation settings: nodes deployment

Each network node carries all the initial data related to the grid where it has been deployed. These data will be loaded from a file. The initial order of these data is random and different for different nodes.

Each node complied with the defined mobility model which has been introduced in Section 4.2.4. The moving speeds for all the nodes are the same which is 2.5 metres per second (2.5 m/s). The pause time for the nodes staying at each destination is 0 second, which means the nodes will start moving again immediately once it reached its destination. The time interval for checking node's current location is 0.1 second.

In this simulation, different nodes have the different start moving times. Consider if all the nodes which were initially deployed in the same Grid start moving at the same time, then the times for all the nodes moving out of their initial Grid are close, because the initial locations for these nodes are the same. Since all the data related to a particular grid are initially being stored on the nodes that are deployed in the Grid which is the same as the data's area of origin, so most of the data will be moving out of its related Grid with their carrying nodes when these nodes move out of their initial Grids. This will cause the data availability being suddenly dropped to a lower value, and then it goes up again because nodes coming from the other Grids are entering this Grid (Appendix A Synchronized Node Start Moving shows the data availability when nodes start moving simultaneously). In order to avoid this happening, the start moving delays of individual nodes that are initially located in the same Grid will be set to different values within a specific time period (start pause time period), thus these nodes will not start moving concurrently. For this simulation, the start pause time period has been set to 0-200 seconds, and the time interval between each node start moving delay is calculated by dividing this start pause time period by number of nodes initially deployed in a Grid which is 16. Figure 4.6 shows the start moving delays for nodes in a single Grid.

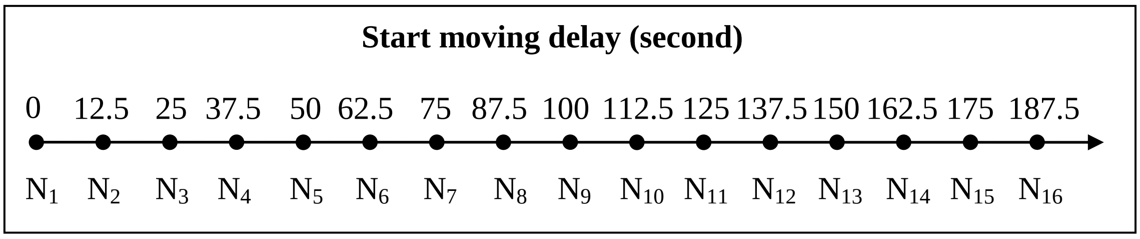


Figure 4.6 Simulation settings: start moving delays of nodes in a single Grid

All the network nodes broadcast data in the same frequency channel. For the basic simulation run, the transmission power of each node set to 0 dBm, and the communication range has been calculated as approximately 50 metres based on the formula which was extracted from Castalia simulator extension with the given transmission power level:

$$\log(\text{TX range}) = ((\text{TXPower}-\max(\text{receiverSensitivity}, \text{noisefloor}+5\text{dBm}))-\text{PL}_{d0}) / (10*\text{pathLossExponent})$$

where TX range is the communication range in metres, TXPower is the transmission power in dBm, and PL_{d0} is the path loss at unit distance d0. The communication range has been calculated based on the following values: 0 dBm for transmission power, -95 dBm for receiver sensitivity, -100 dBm for the noise floor, 54 dBm for the PL_{d0} with 1 metre for d0, and 2.4 for the path loss exponent.

4.2.7 Bandwidth

The data prioritization of the Data Hovering algorithm is essential for improving the quality of retained data if the time for transmitting all the data which are moving out of their area of origin (related data) with their carrying node is limited. In this case, there are three major parameters must be taken into consideration: bandwidth, total size of related data to be transmitted, and the length of the maximum available transmission time for transmitting all the data related to a single Grid. To ensure that there is not enough time to transmit all the related data within limited transmission time (which would be the real-world case where new sampled data is added periodically), the bandwidth must be smaller than the result of dividing total size of related data on a node by the maximum available transmission time. The relationship among these three parameters can be expressed as the following equation:

$$\text{Bandwidth (bits/second)} < \text{Total size of related data on a single node (bits)} / \text{max available transmission time (second)}$$

In particular, the total size of related data on a node is equal to the result of multiplication of the size of one piece of related data and the total number of related data. The value for the total size of related data on a node can be calculated as 20480 bits based on the given values (described in section 4.2.5): 320 bits for size of a single data, and totally 64 data for each Grid.

The maximum available transmission time is the result of dividing maximum transmission path for transmitting all the data related to a single Grid by node moving speed. Based on the when to transmit and what to transmit policies which has been introduced in section 3.4and section 3.5, the node will start transmit the data related to a particular Grid (G') when it is moving out of the Grid (G'), and transmission of data related to the Grid (G') will be stopped when the node moving into any other Grid which is not adjacent to the Grid (G'). In addition, the network nodes are always moving in a straight line as defined in mobility model (section 4.2.4), and therefore, the maximum transmission path for a node to transmit all the data related to a Grid is a diagonal line starts from one corner of a Grid to another opposite corner of its adjacent Grid. Figure 4.7 shows an example of maximum transmission path. In this example, a node is start moving from location A in Grid G4. The node is moving out of G4 at location B, and it starts transmitting the data related to G4. When the node reaching at location C, it stops transmitting the data related to G4, thus the maximum transmission path for transmitting the data related to G4 is between location B and C. As the size of each Grid in this simulation is 100m x 100m, the maximum transmission path can be computed as 223.6 metres ($\sqrt{(100^2 + (100*2)^2)}$). Since the node moving speed has been set to 2.5 m/s, the maximum available transmission time will be 89.44 seconds (223.6 / 2.5).

The maximum value of the bandwidth is approximately 228.98 bits per second (20480 / 89.44) based on the computed total size of related data and maximum transmission time. The bandwidth in this simulation has been set to 200 kb/s, which is smaller than the maximum value, so that the node does not have enough time to transmit all the data related to a single Grid in most of the time.

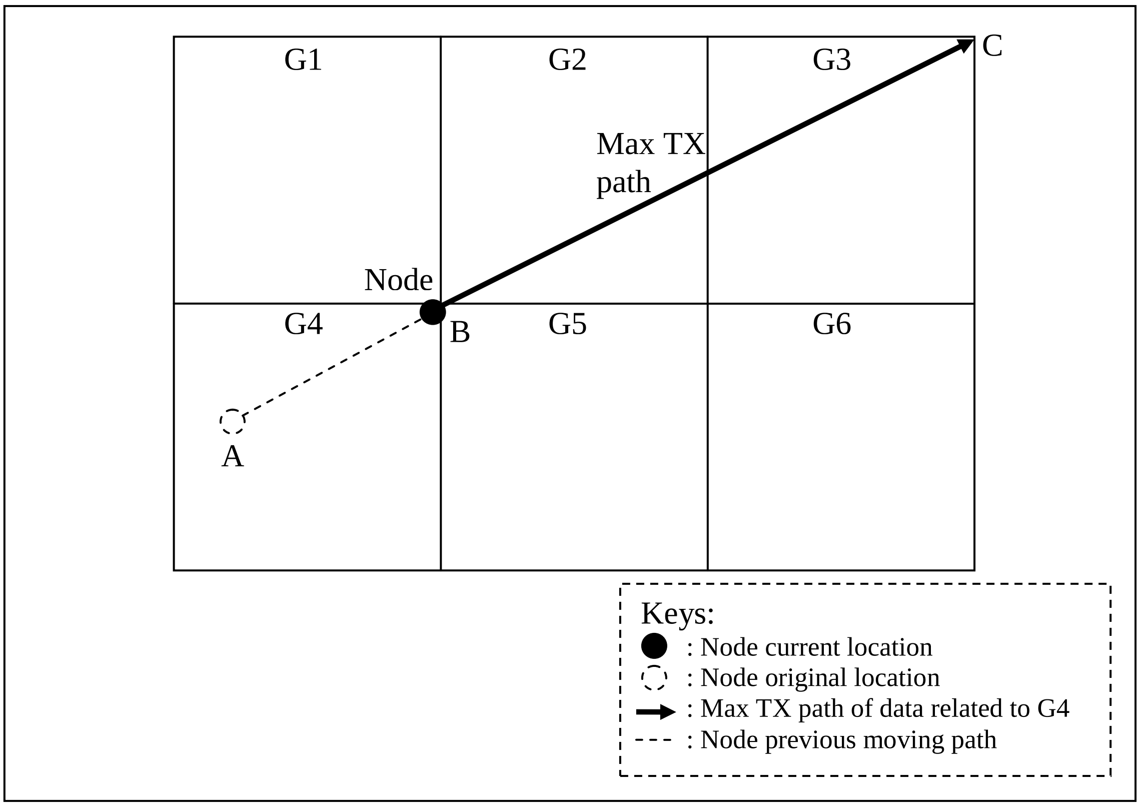

G1
G2
G3
C
Max TX path
Node
G4
B
G5
G6
A
Keys:
● : Node current location
◌ : Node original location
→ : Max TX path of data related to G4
- - - : Node previous moving path

Figure 4.7 Maximum transmission path of data related to a Grid

4.2.8 Garbage Data

There are two requirements which must be fulfilled in this simulation of Data Hovering algorithm:

1) The total number of nodes within the network area should be constant except for the case that a node disconnects from the network if its power level is low. This is because the changing of total number of network nodes will affect the number of data can be retained in their area of origin, so that the data availability would be varied because of the variation of the total number of network nodes rather than affecting by applying Data Hovering algorithm. Therefore, the total number of network nodes should remain unchanged in order to be able to verify how Data Hovering algorithm would affect the data availability (assumes in section 3.2).

2) The total number of data which were sampling from this network area should remain unchanged except for the case that the data is getting lost from this network area with its carrying node, because its carrier's power level is low. If the total number of data will be changed during the simulation, then the number of data related to a Grid will be changed, and the size of total data related to a particular Grid will also be changed. Thus, it cannot be guaranteed that the available time for transmitting all the data related to a particular Grid is limited (as mentioned in section 4.2.7).

If the network nodes can only move in the network area, then the requirement 1 can be satisfied. However, at a certain time after the simulation started, it is possible that each network node will carry all the initial data, as this research assumes that each node has unlimited memory. In this case, the data availability is only affected by the locations of the network nodes. One possible way for addressing this issue is that the node will empty its memory by deleting all its carrying data when reaching any boundary of the network area to simulate a node moving out of the network area forever and another node joining this network area with no knowledge of this area. However, the total number of initial data will be decreased dramatically when nodes reaching the network boundaries. Thus, the Garbage Data concept has been defined in order to satisfy both requirements.

Each initial data has a Boolean parameter: isGarbage. This parameter indicates the garbage status of a node, and it will be initially set to "false". The network node sets the isGarbage parameter of all its carrying data to "true" instead of deleting them when reaching the network boundary, and the sampling times and sampling locations of these data remain unchanged, so that these data can still be transmitted. When a node received a data from another node, it checks the existence by comparing the received data's sampling time and sampling location with the existing data. If none of them has the same sampling time and sampling location, then it inserts the received data based on different data prioritization policies. Otherwise, the received data has a duplicate copy, and then the node will replace the duplicate data only if the isGarbage parameter of the received data is "false" and the duplicate data's is "true". Figure 4.8 shows how Garbage Data works when a node received a data in a flow chart.

In addition, the data which their isGarbage parameter is "true" will not be taken into consideration when calculating the data availability and data quality (see section 4.3.1).

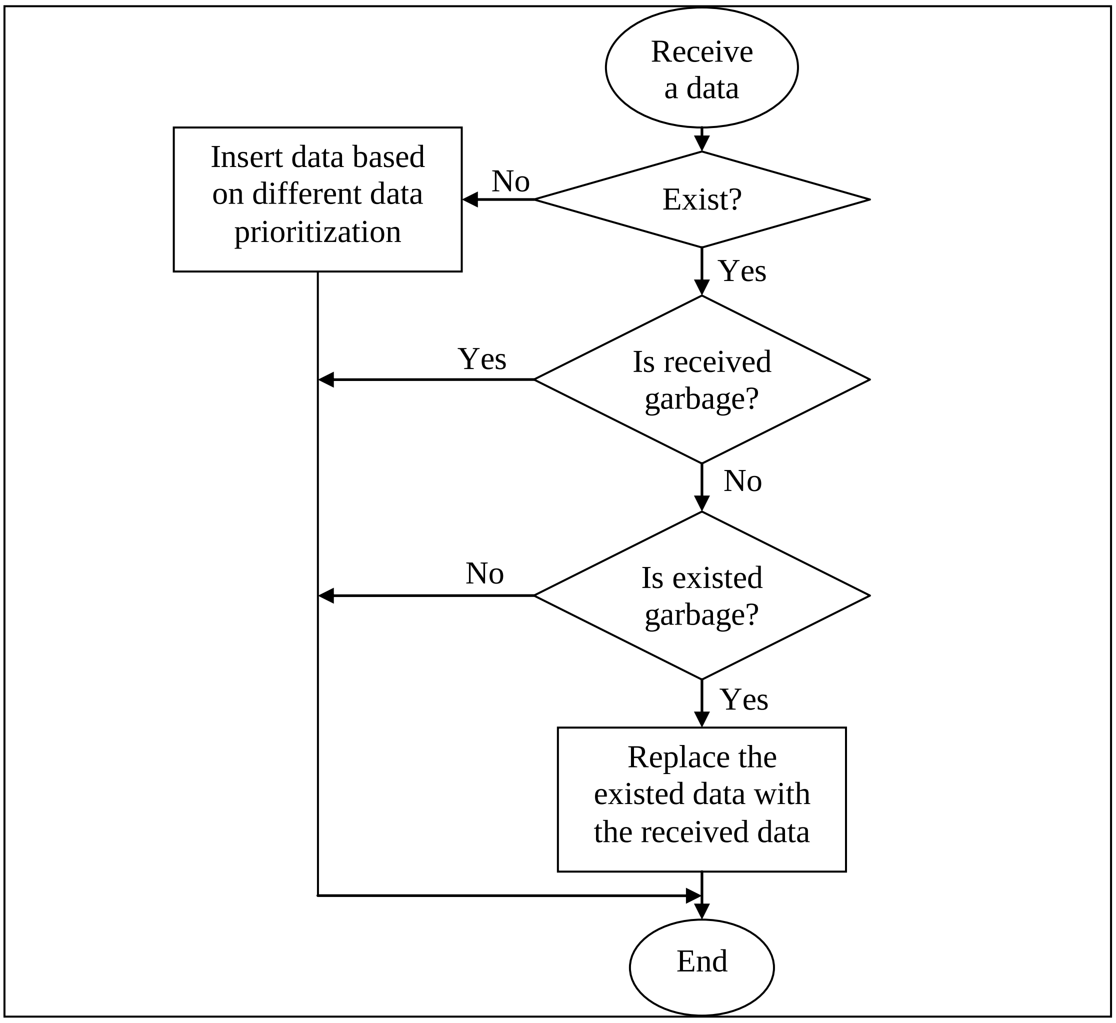

Receive a data
Insert data based on different data prioritization
No
Exist?
Yes
Yes
Is received garbage?
No
No
Is existed garbage?
Yes
Replace the existed data with the received data
End

Figure 4.8 Flow chart of Garbage Data: when a node received a data

4.2.9 Initial Transmission

Chapter 3 has introduced the defined policies of Data Hovering algorithm and the baseline approach which will be used for comparison. In the baseline approach, the nodes start transmitting data immediately when the simulation starts. The "when to transmit" policy of the Data Hovering algorithm defines that the data transmission will not be started until their carrying node moved out of a Grid after simulation starts. Thus, there is a gap for no data transmission for defined policies, but the baseline does not have this time period. It is necessary to make the defined policies of Data Hovering algorithm start transmitting data at the same time as baseline approach, in order to be able to measure the data availability and data quality: the nodes will start transmitting the data related to their initial Grid when the simulation starts for the simulation of defined Data Hovering policies.

4.2.10 Safe and Risk Radius of Hovering Information Algorithm

As introduced in Section 2.3.1, the data in Hovering Information approach [43] is associated with different areas which centred at its sampling location, but with different radii. The node complying with Hovering Information algorithm will start transmission data if its current location is within the area between the risk area and safe area of this data, and it will stop transmitting this data if it is no longer within this data's risk area. For the simulation of Hovering Information algorithm, the radii of the safe area and the risk area must be set. Larger risk area and smaller safe area of data results longer transmission time for this data. This will result higher transmission priority of this data. However, each data in this simulation is unique and the prioritization of the data is through different policies. Thus, different data should have the same safe radius and the same risk radius. Based on the settings of the size of the Grids, the distance between no transmission and start transmission of a data in defined Data Hovering algorithm is varied depending on the movements of the node. This value is varied from 0 to the diagonal distance between two corners of a single Grid which is 141.42 metres. The distance between stop transmission and start transmission is varied from 0 to the diagonal distance from one corner of a Grid to another corner of its adjacent Grid which is 233.61 metres. A medium value has been set for both radii based on these two values: safe radius = 70 metres, and risk radius = 180 metres.

4.2.11 Scenarios

Five scenarios have been designed for this simulation. Each scenario complied with the different Data Hovering policies:

- The baseline
- Hovering Information Broadcasting algorithm
- When to transmit + what to transmit + random prioritization
- When to transmit + what to transmit + random temporal granularity prioritization
- When to transmit + what to transmit + adaptive temporal granularity prioritization

For each scenario, four simulation runs with the simulation settings which have been described in this section and varying the transmission power level will be carried out. The transmission power level for these simulation runs are: 0 dBm, -5 dBm, -10 dBm, and -15 dBm. Each simulation run will last for 3000 seconds in simulation time.

4.2.12 Simulation Settings Review

Table 4.2 shows review of the major simulation settings which have been described in this section.

Network and Grids	
Network size	400m x 400m
Grid size	100m x 100m
No. of Grids	16
Mobility model	
Mobility model	Defined mobility model
Node start moving delay	Vary from 0-200s
Pause time	0s
Node moving speed	2.5m/s
Location check interval	0.1s
Data	
Total No. of data	1024
No. of data related to each Grid	64
Data sampling time period	0-1024s
Size of single data	320bits
Data sampling location	Evenly spread over network area
Data sampling time	Evenly spread over sampling time period
Nodes	
Total No. of nodes	256
No. of nodes in each Grid	16
Node location	Centre of its initial Grid
Node initial carrying data	All data related to its initial Grid
TX power level	0dBm, -5dBm, -10dBm, -15dBm; vary for

	simulation runs
Communication range	≈50m for 0dBm, vary based on different transmission power level
Bandwidth	
Bandwidth	200bits/s
Simulation	
Simulation time limit	3000s

Table 4.2 Review of simulation settings

4.3 Evaluation Methodology

4.3.1 Evaluation Metrics

The performance of Data Hovering algorithm will be evaluated in terms of data availability and data quality. The data quality can either be measured in temporally or spatially. As the defined Data Hovering policies of this research only focus on temporal data prioritization, thus the temporal quality of retained data will be measured.

Data Availability

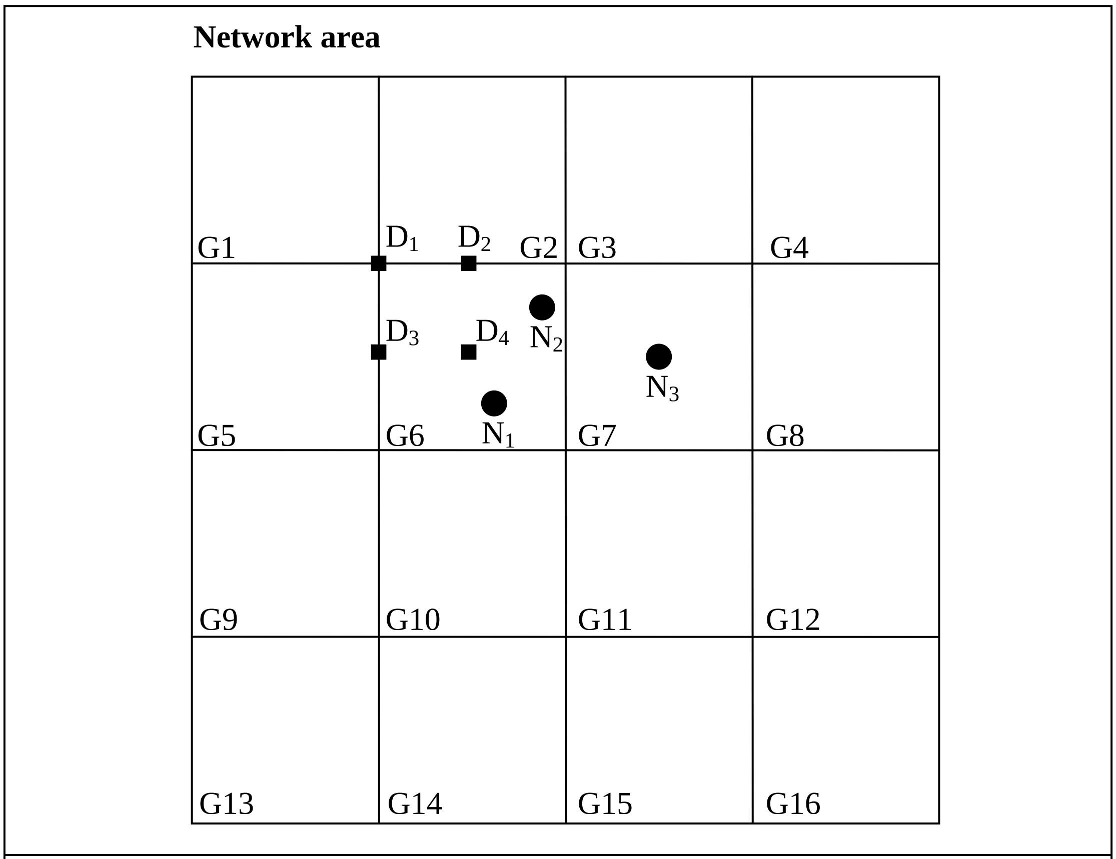
Data availability will be used for measuring how much data related to a particular Grid retains in its area of origin at a certain time. The higher value of data availability indicates more data have been retained in their area of origin.

A piece of data is thought to be available if its garbage status is not true and the current Grid of its carrying node is the same as the Grid of its area of origin. Thus, the data availability of a single Grid is equal to the total number of non-duplicate available data of this Grid divided by total number of non-duplicate initial data related to this Grid.

To compute the total number of non-duplicate available data of a specific Grid at a given time, all the non-garbage data carrying by nodes will be temporarily stored in an array if the current Grid of their carrying node is the same as this specific Grid and the sampling Grid of these data is also the same as this specific Grid. The duplicate data will then be removed from this array if any other data in this array has the same sampling time and sampling location. The total number of data of this array represents the total number of non-duplicate available data. The temporary array will be emptied before every computation of the data availability.

Data availability of a Grid (%) = total number of non-duplicate available data of this Grid / total number of non-duplicate initial data related to this Grid

Figure 4.9 shows an example of available data. There are four initial data related to Grid G6: D_1, D_2, D_3, and D_4. Three nodes are currently in the network area: N_1 and N_2 are currently located in G6 and N_3 is currently located in G7. The data currently carrying by N_1 are D_1 and D_2, D_2 and D_3 are carrying by N_2, and D_4 is currently carrying by N_3. As a data is available if the current Grid of its carrying node is the same as its sampling Grid, the available data of Grid G6 are D_1, D_2 and D_3. Thus, the data availability of G6 in this example is 75%.



Nodes	Current Grid	Carrying data
N_1	G6	D_1, D_2
N_2	G6	D_2, D_3
N_3	G7	D_4

Data	Sampling Grid
D_1	G6
D_2	G6
D_3	G6
D_4	G6

Available data of G6
D_1
D_2
D_3

Figure 4.9 Example of available data

Temporal Data Quality

Temporal data quality will be used for measuring how well do the available data of a Grid spread over the sampling time period at a certain time. The following steps will be carried out for calculating the temporal data quality of a Grid at given time:

1) Calculate the average number (**avgTD**) of the time difference (**TD**) between the sampling time of each temporally adjacent available data and the time difference between the available data which has latest sampling time and the maximum of the sampling time period. If the sampling time of any available data is the same as the minimum time of the sampling time period (based on the introduction of the initial data described in section 4.2.5), then the time difference between the minimum of the sampling time period and the available data with earliest sampling time will also be involved in the calculation of avgTD. An available data is temporally adjacent to another one if their sampling times are adjacent to each other. For example, if there are three available data (as shown in Figure 4.10): D_1, D_2, and D_3, and assume the sampling times for these data are 0 second, 5 seconds and 10 seconds respectively, then D_1 is temporally adjacent to D_2, D_2 is temporally adjacent to D_3, but D_1 is not temporally adjacent to D_3. The average sampling time difference will be calculated by dividing the length of sampling time period by the number of time differences of the available data. The length of sampling time period can be computed by subtracting the minimum from the maximum of the sampling time period. The number of time differences is equal to the sum of total number of available data in a given Grid and 1. However, if any of the available data has the same sampling time as the minimum of the sampling time period, the number of time differences is then equal to the total number of available data.

$$\text{avgTD} = (\text{MaxT} - \text{MinT}) / \text{total number of TD}$$

where MaxT and MinT are the maximum and minimum value of the sampling time period respectively. Total number of TD = total number of data + 1 or total number of data if any available data has the same sampling time as MinT.

2) Calculate the standard deviation (**SD**) of the time differences (TD) with a given average (avgTD) of the time differences, and this average has already been calculated in last step.

$$SD = \sqrt{((\sum (TD - avgTD)^2) / \text{total number of TD})}$$

An example of calculating the standard deviation of the time differences of available data has shown in Figure 4.10: $SD = \sqrt{(((TD_1 - avgTD)^2 + (TD_2 - avgTD)^2 + (TD_3 - avgTD)^2) / 3)}$.

3) Calculate the maximum value of the standard deviation (**maxSD**). To maximize the value of the standard deviation, the difference between each TD and avgTD must be maximized. Since the value of avgTD is fixed, so that the value of TD must be maximized. For this purpose, the sampling times of all the available data should be the same as maximum of the sampling time period. Thus, the maximum value of each TD is equal to the difference between maximum (**MaxT**) and the minimum (**MinT**) of the sampling time period. Therefore, the maxSD can be computed as the following equation:

$$\begin{aligned} maxSD &= \sqrt{((MaxT - MinT - avgTD)^2 * \text{total number of TD} / \text{total number of TD})} \\ &= MaxT - MinT - avgTD \end{aligned}$$

4) Calculate the temporal data quality:

$$\text{Temporal data quality (\%)} = 1 - (SD / maxSD)$$

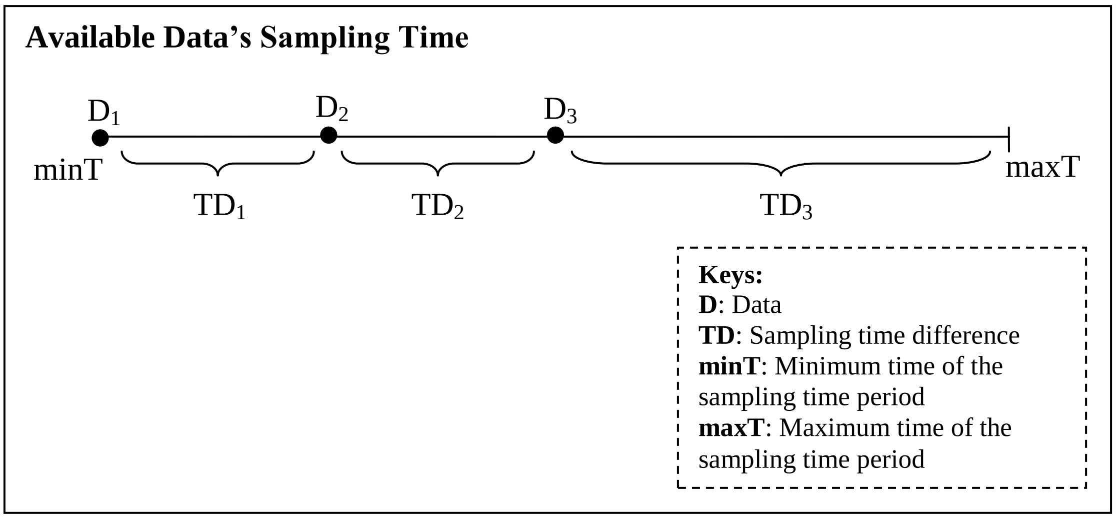

Available Data’s Sampling Time
D1
D2
D3
minT
maxT
TD1
TD2
TD3
Keys:
D: Data
TD: Sampling time difference
minT: Minimum time of the sampling time period
maxT: Maximum time of the sampling time period

Figure 4.10 Calculate the standard deviation of the time differences of the available data

4.3.2 Gathering Statistics

The data availability and the temporal data quality will be periodically computed for the centre four Grids of the network area during the simulation, and the time interval for every computation is 1 second simulation time. All the computed data availabilities and temporal data qualities will be output to a file when each simulation run completed.

4.3.3 Generating Outputs

To simulate an infinite network, Grids along the edge of the simulation area would not have the typical performance due to the effect of the proximity of the boundary where retained data is discarded. Thus, the data availabilities and temporal data qualities will be measured for the centre four Grids of the network area. The average value of data availabilities and temporal data qualities of these four Grids at each computation time interval will then be calculated. The evaluation metrics for Data Hovering algorithm are data availability and temporal data quality, so it is necessary to compute their product to express the performance of Data Hovering algorithm.

Graphs of data availability and spatial data quality at different transmission power level will be plotted for analysis of the simulation results and comparison of the defined

Data Hovering algorithm with the baseline. In order to reduce the noise of the graphs, the moving average will be applied on the set of simulation results (e.g. average data availabilities of the centre 4 Grids of a simulation run), where the size of the fixed subset of the moving average sets to 200.

Chapter 5

Evaluation

5.1 Overview

Different “Data Hovering” algorithms and “Hovering Information broadcast” algorithm have been simulated with the settings discussed in Chapter 4. Each “Data Hovering” algorithm complies with different Data Hovering policies. This chapter describes the simulation results, and follows by an analysis of their behaviours. In order to easily distinguish different algorithms, the following abbreviations of the combination of the policies will be used in this chapter (see Table 5.1).

Abbreviation	**Data Hovering algorithm complying with different policies**
Baseline	Baseline
Random	When to transmit + what to transmit + random prioritization
RTG	When to transmit + what to transmit + random temporal granularity prioritization
Adaptive	When to transmit + what to transmit + adaptive temporal granularity prioritization
HoverInfo	Hovering Information Broadcasting algorithm

Table 5.1 Abbreviations for Data Hovering algorithms complying with different policies

5.2 Data Availability

5.2.1 Overview

The average data availability of the centre four Grids over the simulation time of the different Data Hovering algorithms with 0dBm transmission power is illustrated in Figure 5.1. As expected, the data availabilities of all of Random, RTG and Adaptive are always higher than the Baseline. This is because the nodes of the Baseline kept transmitting the random data without considering the current location of the node and the sampling location of the data. With the proposed algorithms complying with when to transmit and what to transmit policies, the nodes only transmitted the relevant data, so that other nodes can receive more data related to the Grid that they are moving towards or currently located.

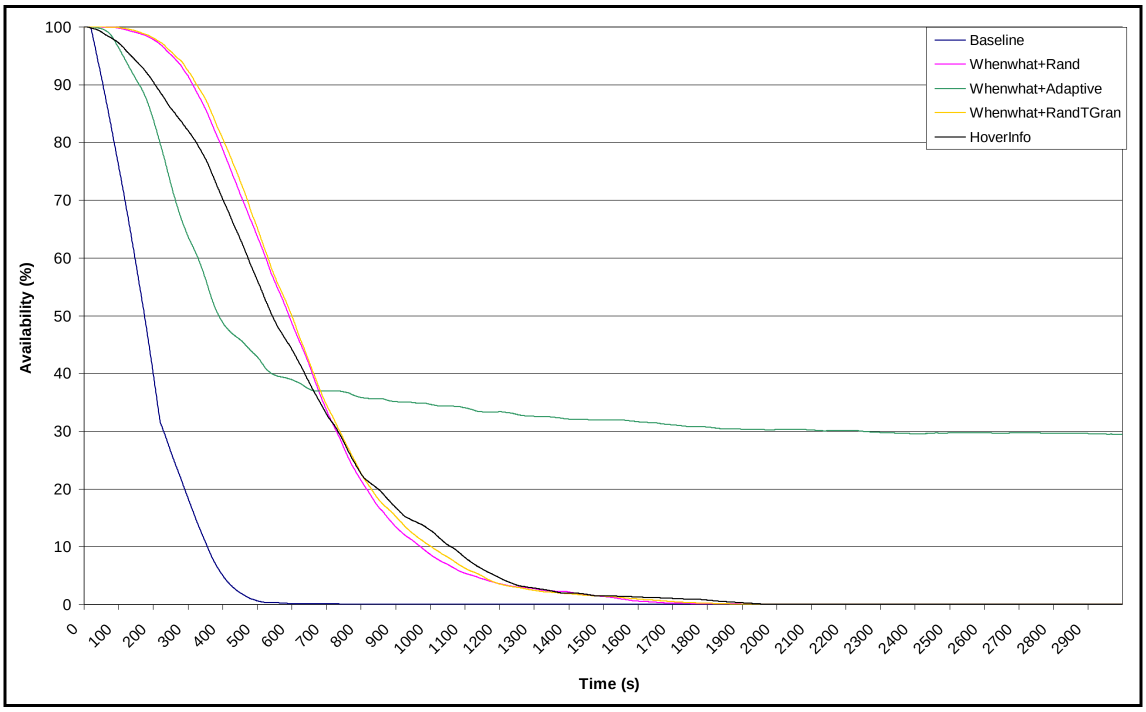


Figure 5.1 Data availability at transmission power 0 dBm

5.2.2 The RTG

Complying with the same “when to transmit” and “what to transmit” policies, the value of the data availability of a Grid will be influenced by the number of duplicate data transmitted by different nodes which have the same previous leaving Grids, when the data transmission time is limited. Higher the number of duplicate data results the lower the data availability and vice versa. Thus, it is likely that a policy will have lower data availability if it has the higher probability having different nodes transmitting the same data.

In this figure (Figure 5.1), the data availability of RTG is almost the same as the Random, except the period between 700 and 1200 seconds where RTG is higher than the Random with maximum 1%. In the RTG approach, each node transmits the data taken from different time segments with higher priority, and the rest of data which have the same time segments as transmitted data have the lower priority. Moreover, the total number of time segments are is less than the total number of initial data related to a single Grid, so that there is at least one piece of data belonging to one single time segment. Thus, each of them first transmits a subset of the relevant data, where the number of data in this subset is less than the total number of its carrying relevant data. Comparing with the Random approach, nodes complying with RTG are likely to transmit more duplicated data when the data transmission time is limited. This can be explained by an example. Considering two nodes (N_1 and N_2) transmitting 4 pieces of data (D_1, D_2, D_3 and D_4). These data are related to the same Grid (G_1), and their sampling times are evenly spread the sampling time period. In RTG approach, the number of time segments will be 2 with 2 pieces of data in each time segment. Suppose the transmission time allows 2 pieces data to be transmitted. The probability of transmitting 2 duplicate data for Random is $(1/4 * 1/3)^2 = 1/144$, and the probability for RTG is $(1/4 * 1/2)^2 = 1/64$. Moreover, each node in RTG would have different number of time segments, so the nodes were not transmitting the same subset of data. This ensures more different data can be transmitted whilst the data from different time segments still have the higher priority. This leads the minor difference between data availabilities of Rand and RTG.

5.2.3 The Adaptive

This figure (Figure 5.1) also suggests that the Adaptive approach performs better than the Random approach in terms of data availability over a long time period. An interesting phenomenon of this figure is that the data availability of the Adaptive is lower than the Random approach before approximately 650 seconds, but it then tends to be stable at 30% and the data availability of the Random approach still keeps decreasing. As described in Section 3.8, the priority for transmitting the data related to a Grid is the same as the data storing position of Grid_Data. In random prioritization (described in Section 3.8.2), the transmitting priorities of data are likely to be different for different nodes which are transmitting the data related to the same Grid, because the storing positions of the data are purely random. Thus, it is pseudo-cooperative, and the nodes that are coming into this Grid would receive more different related data. In "adaptive temporal granularity prioritization" (Section 3.8.4), the time segment indices are fixed once they have been built, and data will be inserted to its appropriate position corresponding to its computed time segment index. Since the initial data are the same for the nodes which are deploying in the same Grid and no more data will be created, the transmitting priorities of data related to the same Grid are the same for all the nodes in this experiment. Therefore, compared with the Random approach, within the limited transmission time, less different related data can be transmitted by nodes complying with the Adaptive algorithm. The data availability of the Adaptive approach is hence lower than the Random approach during the early stage of the simulation. However, the total number of initial data was decreasing whilst the simulation time was passing, because the network nodes set their carrying data to be garbage when reaching any network boundary. With adaptive prioritization, data with a higher transmission priority could always be transmitted within the limited transmission time, so it is likely that these data would always be retained in their original Grids. This leads to the decrease of data availability of the Adaptive approach to be slowed down.

As discussed in the above paragraph, the steep rate of decay of availability in the Adaptive is caused by the same data transmission priority in different nodes and it results more duplicate data to be transmitted by different nodes. There are two possible improvements can be made to reduce the number of duplicate data to be transmitted:

1. Randomized the prioritization of data for individual nodes, thus different nodes will transmit the relevant data in different orders.
2. The nodes transmit their data through collaboration to enable different nodes transmit the different subset of data within the limited transmission time (see future work in Section 6.2.4).

For the first improvement, the randomization process can be performed during the process of constructing the time segment indices which has been discussed in Section 3.8.4. The current prioritization algorithm continuously divides the sampling time period into 2^n groups and inserts the smallest time segment index from each group into the list of prioritized time segment indices. This can be varied by inserting the random time segment index instead of the smallest one which was taken from each group into the list of prioritized list. This improvement would results the less duplicate data related to the same Grid being transmitted by different nodes, since different nodes will have different data transmission order. In terms of temporal data quality, this algorithm still eliminates some possibilities of achieving lower temporal data quality when comparing with the Random. Thus, this improvement should outperform the Random in terms of temporal data quality, even though its quality might be lower than the current Adaptive prioritization without the randomization.

5.2.4 The HoverInfo

In Figure 5.1, the data availability of the HoverInfo is lower than the Random before approximately 700 seconds, and it is better than the Random with a maximum 4% between 700 to 1250 seconds. In this simulation, each node initially carries the data related to the Grid of its initial location. Unlike the other defined Data Hovering policies where the node transmits all the data related to its previous entering Grids, the node complying with the HoverInfo transmits the data when it is located in the area between the risk area and the safe area of this data. In this case, at the beginning stage of the simulation, each node complying with the HoverInfo transmits a subset of data related to its initial Grid. Comparing with the Random approach, it has higher probability to transmit more duplicate data. This results the lower data availability of the HoverInfo before 700 seconds. This is the initialization process of the HoverInfo based on current simulation settings, and it can be reduced by adjusting the simulation settings.

The nodes travelling inside the network area will receive data from the other nodes. They will then carry the data not only related to their current Grid but also related to other Grids. In the HoverInfo, the data availability will be influenced by the moving paths of the nodes. Similar moving paths of different nodes result less number of different data to be transmitted. Thus, it lowers the data availability, and otherwise vice versa. In addition, the moving paths of the receiving nodes also affect the data availability. In the Random approach, data related to previous leaving Grid of its carrying node always have higher transmission priority than the data related to other Grids. In contrast to the Random approach, the node complying with the HoverInfo transmits the data related to its current Grid and possibly the data related to the adjacent Grids simultaneously. There are four possible situations may arise which would achieve different data availabilities:

1. If all the nodes, which have received data from the sender, are moving towards the Grid that the sender left, then the data availability of the HoverInfo would be lower than the Random. This is because, with the HoverInfo, the number of transmitted data related to the previous leaving Grid of the sender is likely to be less during the limited transmission time.
2. If all the receiving nodes are moving towards the adjacent Grids of the previous leaving Grid of the sender, then the data availability of the HoverInfo would be higher than the Random, because of the lower probability for transmitting the data related to other Grids by nodes complied with the Random.
3. If all the receiving nodes are moving towards the previous leaving Grid and the adjacent Grids, then the average data availability of these Grids would tends to be the same for both approaches. In addition, this would be varied depending on the number of data received by the receiving nodes.

4. If all the receiving nodes are moving towards the other Grids in which none of the transmitted data are related to these Grids, then the data availability of both approaches would tends to be the same.

5.2.5 Lower Transmission Power

Figure 5.2, Figure 5.3 and Figure 5.4 depict the average data availability of the centre four Grids with different transmission powers over the simulation time for different Data Hovering algorithms. These figures indicate that as the transmission power decreases, the difference between the data availabilities of the different Data Hovering algorithms decreases. As described in Section 4.2.6, the communication range is directly proportional to the transmission power. Since the number of nodes within the communication range will be decreased when the communication range is decreased, less nodes could receive the transmitted data from the sender. This leads the decrease of data availability to be faster when reducing the transmission power level.

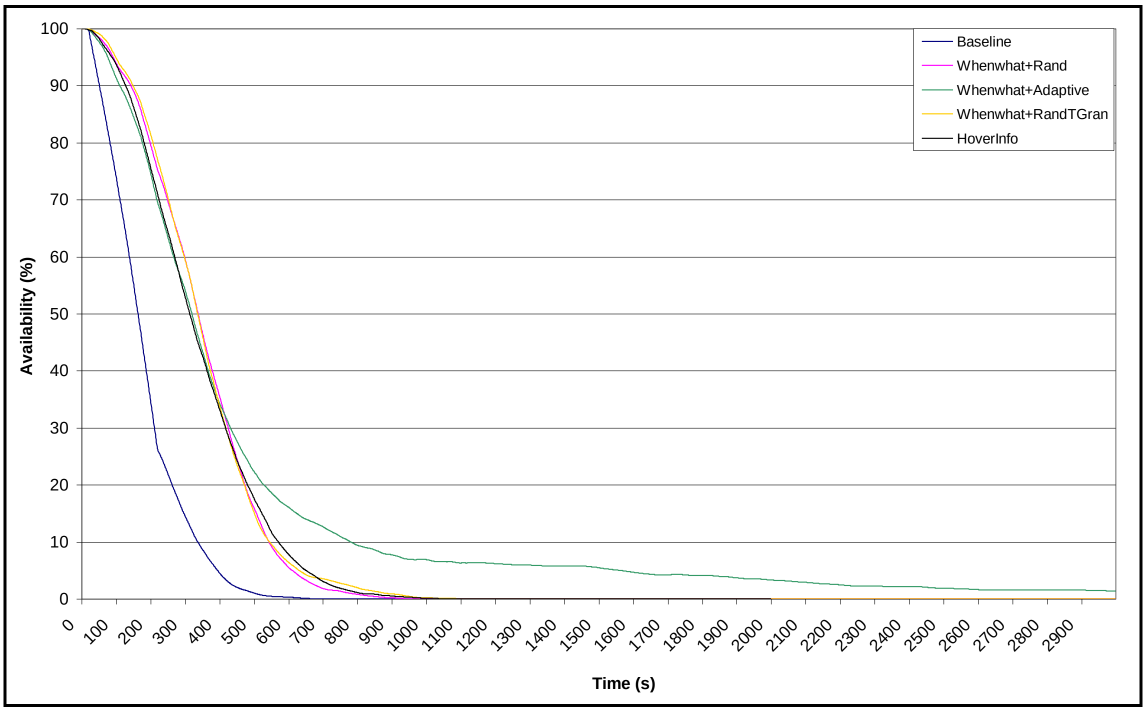


Figure 5.2 Data availability at transmission power -5 dBm

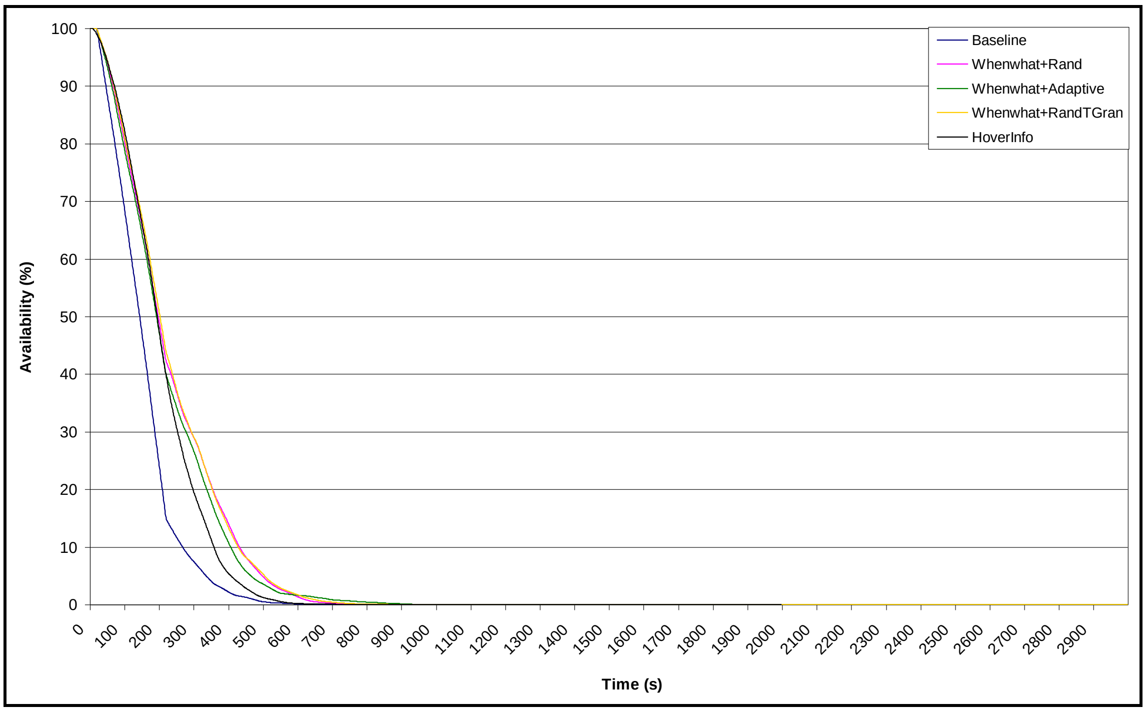


Figure 5.3 Data availability at transmission power -10 dBm

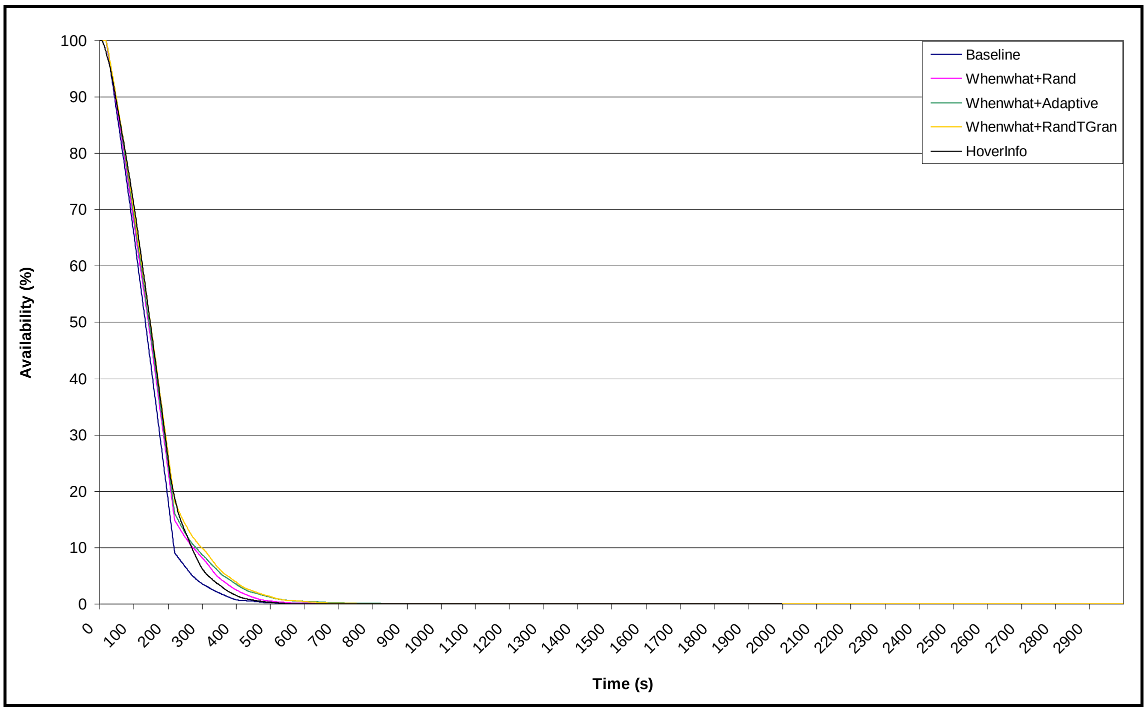


Figure 5.4 Data availability at transmission power -15 dBm

5.3 Temporal Data Quality

5.3.1 The Random

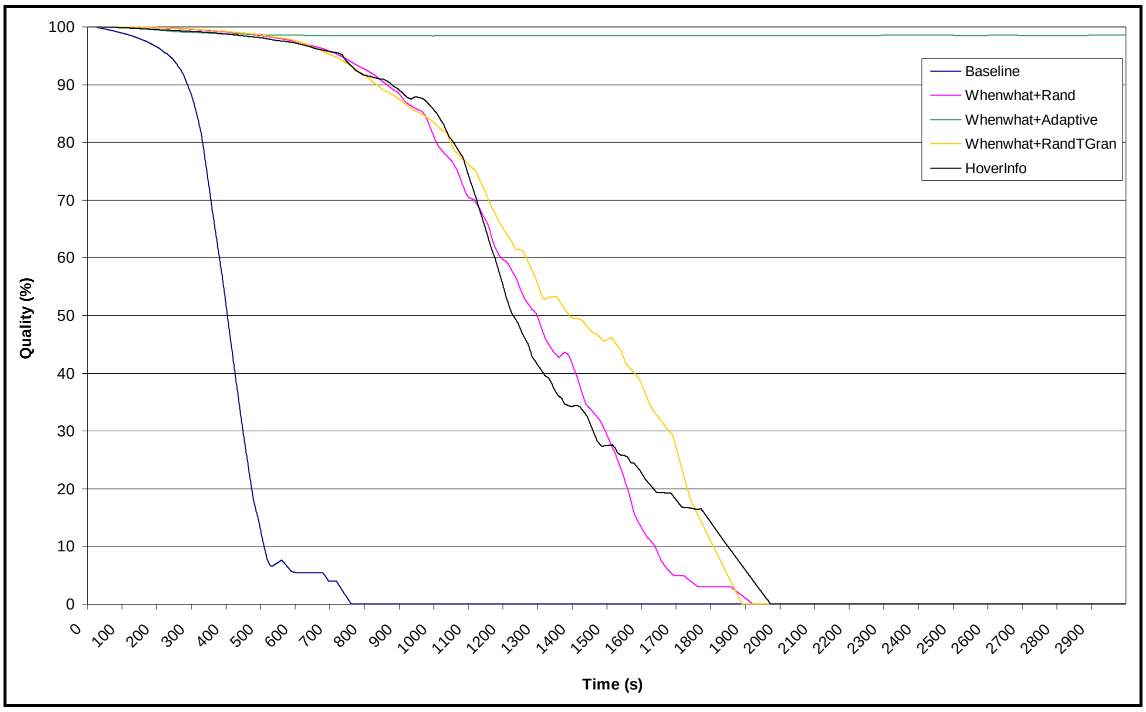


Figure 5.5 Temporal data quality at transmission power 0 dBm

Figure 5.5 presents the average temporal data quality of the centre four Grids for different Data Hovering algorithms at 0 dBm of transmission power. The shape of the curves for the Baseline and the Random are very similar, but their slopes are different. The temporal data quality of the Baseline slowly decreases before approximately 300 seconds, and then it drops to a lower value, and it reaches 0 at approximately 850 seconds. The Random approach took approximately 900 seconds before its temporal data quality starts faster decreasing. During the time period between 1350 seconds and 1550 seconds, the curve of the temporal data quality for the Random approach is not smooth. This is because with the simulation settings described in Section 4.2, when the data availability is lower, the difference between the minimum and maximum (100%)

possible data qualities is higher. In this case, one more data getting lost from its area of origin would have a significant impact, either positive or negative, on its temporal data quality.

The Random approach outperforms the Baseline in terms of temporal data quality, because of the relationship between the data availability and the temporal data quality. Since the data transmission priority for both algorithms is random, if there are more data have been retained in their area of origin, and then it has the higher probability to have higher temporal data quality. The data availability of the Random approach is higher than the Baseline at any given time which has been shown in Figure 5.1, so that the temporal data quality of the Random approach is higher than the Baseline.

The relationship between data availability and temporal data quality can be explained by the following example. Suppose there are 8 initial data which have been taken from the same Grid. The sampling times of these data are evenly spread over a certain sampling time period. Assume that the time difference between any 2 data, whose sampling times are adjacent to each other, is 1 second. We use two instances to describe this relationship: A. 7 data have been retained in their area of origin (i.e. 1 data lost), and B. 6 data have been retained in their area of origin (i.e. 2 data lost). Depending on which data was getting lost, 2 different values of the temporal data quality in different circumstances can be computed for instance A based on the formula which has been introduced in section 4.3.1. Figure 5.6 shows the examples of which data were getting lost from their area of origin for this instance. A1 and A2 in this figure represent the following circumstances, respectively.

A1. 1 data was getting lost where its sampling time is the same as the smallest value of the sampling time period. The temporal data quality for this circumstance will be 100%, and the probability that this circumstance will occur is 1/8.

$Q_{A1} = 100\%$

$P_{A1} = 1/8$

A2. Any other data was getting lost except the data that was sampling at the earliest of the sampling time period. The computed temporal data quality for this circumstance is 94.90%, and its probability is 7/8.

Q_{A2} = 94.90%

P_{A2} = 7/8

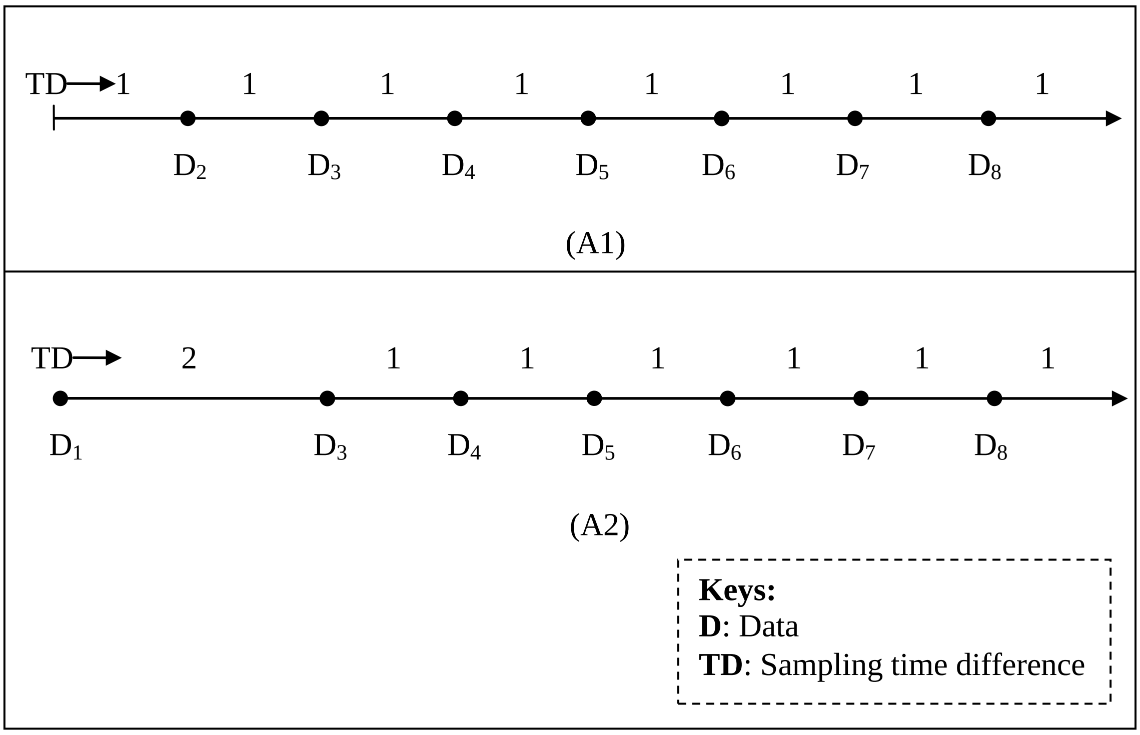


Figure 5.6 Example of which data has been lost when 1 out of 8 data was getting lost from its area of origin

There are 3 circumstances with different values of temporal data quality for instance B. Figure 5.7 shows the examples of which data were getting lost from their area of origin when 6 data have been retained, and B1, B2 and B3 represent the following circumstances, respectively.

B1. 2 data were getting lost, in which one of the data's sampling time is the same as the smallest value of the sampling time period. For instance, D_1 and D_2 were getting lost. The time differences between the sampling times of temporally adjacent data are 2, 1, 1, 1, 1, 1 and 1. The temporal data quality for this circumstance is 94.90%. The probability that this circumstance will occur is 7/28, since there are 7 possible data permutations for satisfying the criteria of this circumstance and the total number of data permutation for instance B is 28.

Q_{B1} = 94.90%

P_{B1} = 7/28

B2. None of the losing data was sampling at the earliest sampling time of the sampling time period, and the sampling times of these 2 data are adjacent to each other. For instance, D_2 and D_3 were getting lost. The temporal data quality for this circumstance is 88.82%. The probability that this circumstance will occur is 6/28.

Q_{B2} = 88.82%

P_{B2} = 6/28

B3. None of the losing data was sampling at the earliest sampling time of the sampling time period, and the sampling times of these 2 data are not adjacent. For instance, D_2 and D_4 were getting lost. The temporal data quality for this circumstance is 92.93%. The probability that this circumstance will occur is 15/28.

Q_{B3} = 92.93%

P_{B3} = 15/28

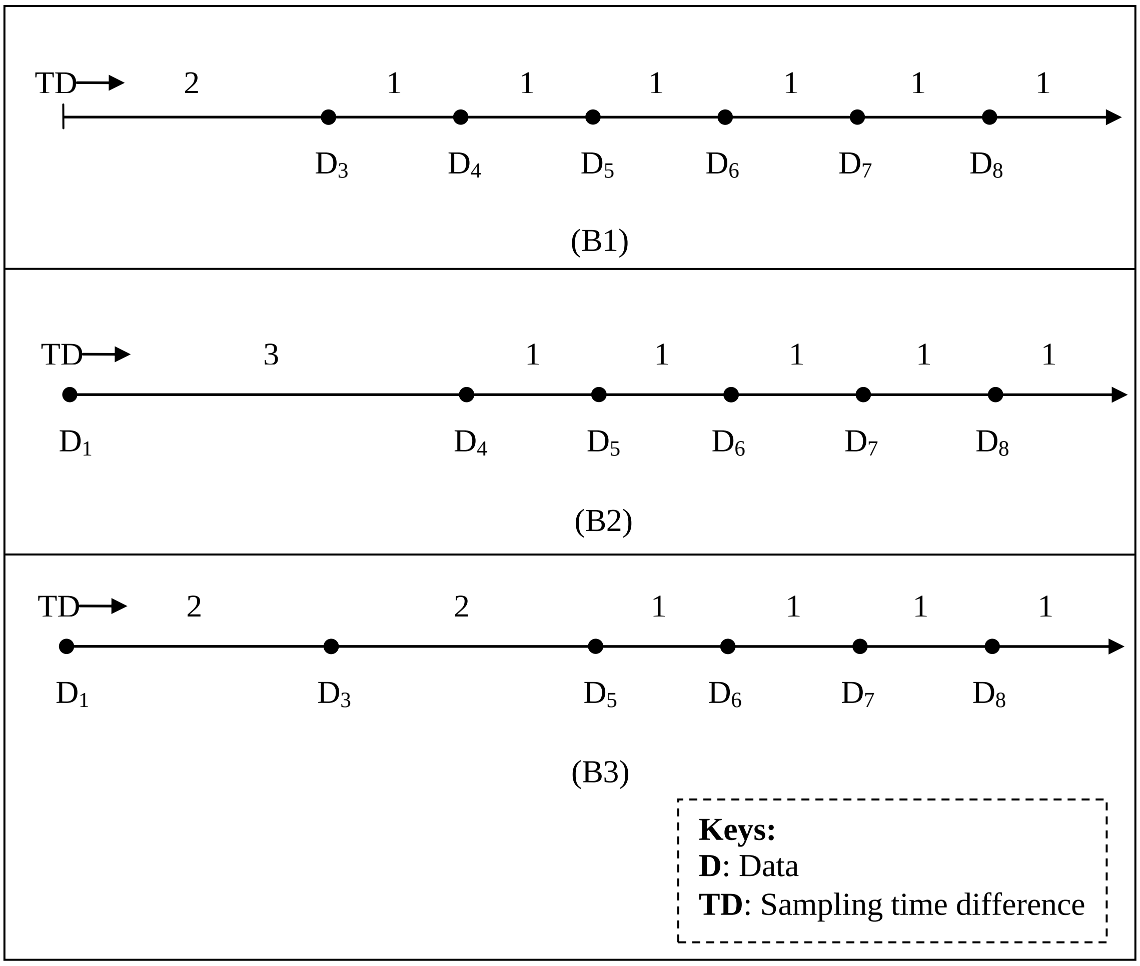


Figure 5.7 Example of which data has been lost when 2 out of 8 data were getting lost from their area of origin

In this example, when there is one piece of data getting lost from its area of origin, the temporal data quality is likely to be 94.90% (A2), because it has the highest probability. It is possible that the temporal data quality could also be 94.90% (B1) when 2 data were getting lost from their area of origin, but the probability associating with B1 is much lower than B3. Thus, the temporal data quality of B is more likely to be 92.93%. This example verifies that higher data availability would possibly lead to the higher temporal data quality.

5.3.2 The RTG

In Figure 5.5, the RTG approach has approximately the same level of temporal data quality as the Random approach before 950 seconds. It then outperforms the Random approach. In RTG, each node groups the data based on its own time segments and the sampling times of the data. The total number of time segments of a single node is less than the total number of initial data, so that there is more than one piece of data in each time segment. As introduced in section 3.8.3, it forces the data to be transmitted from different time segments unless no more data belong to the different time segments. Comparing with the Random approach, each node RTG eliminates some possibilities of having temporally adjacent data to be consecutively transmitted. Since the temporal data quality is higher when the retained data are spread the sampling time period, RTG results higher probability to have higher quality. In case of transmitting the data by more than one node, RTG still have the higher probability to achieve the higher quality if the number of data transmitted by each node is more than 1 and less than the number of its own time segments, because some possibilities of retained temporal adjacent data will still be eliminated. However, the difference between these probabilities becomes smaller when the number of transmitted data by each node increased, because RTG does not prioritize the data in G_Duplicate. This can be explained by the following example in next paragraph.

Consider there are 4 pieces of data (D_1, D_2, D_3 and D_4) which were initially carrying by a single node. Assume these data are evenly spread the sampling time period with a difference of 1 second between sampling times of temporal adjacent data (as shown in Figure 5.8). If the transmission time allows 2 pieces of data can be transmitted and assumes these 2 pieces of data retained in their area of origin, then the qualities of different retained data associating with their probabilities of the Random and RTG approaches are shown in Table 5.2 and Table 5.3 respectively. When there are 4 pieces of initial data on a single node, the total number of time segments of a RTG node is 2, so that there are 2 pieces of data in each time segment. A node complying with RTG eliminates the possibility of transmitting D_1 and D_2, and D_3 and D_4. In this case, the probability of having 82.32% quality for RTG is 0.5 which has the highest probability, and it is the same as the Random approach. However, its probability for achieving the highest quality (0.25) is higher than the Random (0.17), and the probability for achieving the lowest quality (0.25) is lower than the Random (0.33). Therefore, the RTG approach has higher probability to have higher quality when the number of data can be transmitted is greater than 1 and less or equal to the number of its time segments. Table 5.4 and Table 5.5 show the possible qualities associating with their probabilities when there are 3 pieces of data can be transmitted. In this case, the Random and RTG approaches are likely to have the same probabilities for reaching the same qualities.

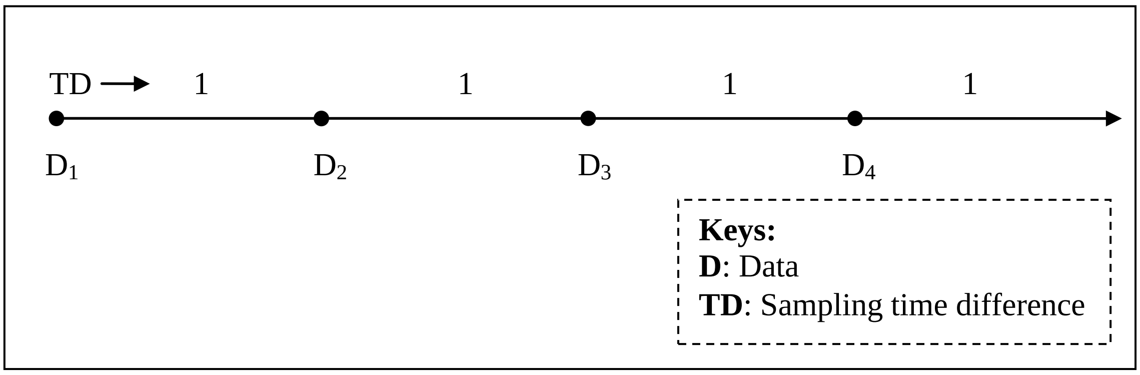


Figure 5.8 Sampling times of initial data of example for evaluating the data quality of RTG

Combination of retained data	Permutations of retained data of this combination	Time differences between temporal adjacent data (sec)	Quality	Probability
D_1, D_2	D_1, D_2 & D_2, D_1	1, 3	50%	4/12 = 0.33
D_1, D_4	D_1, D_4 & D_4, D_1			
D_1, D_3	D_1, D_3 & D_3, D_1	2, 2	100%	2/12 = 0.17
D_2, D_3	D_2, D_3 & D_3, D_2	1, 1, 2	82.32%	6/12 = 0.5
D_2, D_4	D_2, D_4 & D_4, D_2			
D_3, D_4	D_3, D_4 & D_4, D_3			

Table 5.2 Probabilities of achieving possible temporal data qualities by Random approach when 2 data transmitted

Combination of retained data	Permutations of retained data of this combination	Time differences between temporal adjacent data (sec)	Quality	Probability
D_1, D_4	D_1, D_4 & D_4, D_1	1, 3	50%	2/8 = 0.25
D_1, D_3	D_1, D_3 & D_3, D_1	2, 2	100%	2/8 = 0.25
D_2, D_3	D_2, D_3 & D_3, D_2	1, 1, 2	82.32%	4/8 = 0.5
D_2, D_4	D_2, D_4 & D_4, D_2			

Table 5.3 Probabilities of achieving possible temporal data qualities by RTG approach when 2 data transmitted

Combination of retained data	Number of permutations of retained of this data combination	Time differences between temporal adjacent data (sec)	Quality	Probability
D_1, D_2, D_3	6	1, 1, 2	82.32%	18/24 = 0.75
D_1, D_2, D_4	6			
D_1, D_3, D_4	6			
D_2, D_3, D_4	6	1, 1, 1, 1	100%	6/24 = 0.25

Table 5.4 Probabilities of achieving possible temporal data qualities by Random approach when 3 data transmitted

Combination of retained data	Number of permutations of retained data of this combination	Time differences between temporal adjacent data (sec)	Quality	Probability
D_1, D_2, D_3	4	1, 1, 2	82.32%	12/16 = 0.75
D_1, D_2, D_4	4			
D_1, D_3, D_4	4			
D_2, D_3, D_4	4	1, 1, 1, 1	100%	4/16 = 0.25

Table 5.5 Probabilities of achieving possible temporal data qualities by RTG approach when 3 data transmitted

5.3.3 The Adaptive

In Figure 5.5, the temporal data quality of the Adaptive approach slowly decreases and it remains at approximately 98% till the end of the simulation. It outperforms the other approaches. As mentioned in section 5.2, the data availability of the Adaptive approach is lower than the Random approach before 650 seconds. The temporal data quality of the Adaptive approach is approximately 0.5% lower than the Random approach before 450 seconds. It outperforms the Random approach after 450 seconds, even its data availability is lower between 450 and 650 seconds. This is because each node in the Adaptive attempting to maximize the temporal data quality for each pair of the transmitted data. This proves that the adaptive temporal granularity prioritization of the transmitting data can effectively improve the temporal data quality, and hence the retained data are spread over the sampling time period as much as possible.

5.3.4 The HoverInfo

The temporal data quality of the HoverInfo is approximately the same as the Random and the RTG before 700 seconds, and then it tends to be unstable. It is higher than the

Random between 870 to 1120 seconds with a maximum 6% and it is lower between 1120 and 1510 seconds with up to 9%. It is also worse than the RTG after 1000 seconds. Since there is no prioritization policy have been defined for the HoverInfo to ensure the retained data of a single Grid spread out the sampling time period, its temporal data quality is expected to be similar to the Random approach. The data transmission priority of a node complying with the HoverInfo is based on its current location and the sampling location of its carrying data. The node with different moving path in a Grid will result different data to be transmitted with different transmission priority. Based on the settings of this simulation, data sampled in the same Grid are associated with the different sampling times. The moving paths of individual nodes influence the temporal data quality. This leads to the variations of temporal data quality in Figure 5.5.

5.3.5 Lower Transmission Power

The temporal data qualities with different transmission powers for these 3 Data Hovering algorithms are illustrated in Figure 5.9, Figure 5.10 and Figure 5.11. These figures show that the difference between these data qualities become smaller when the transmission power decreased.

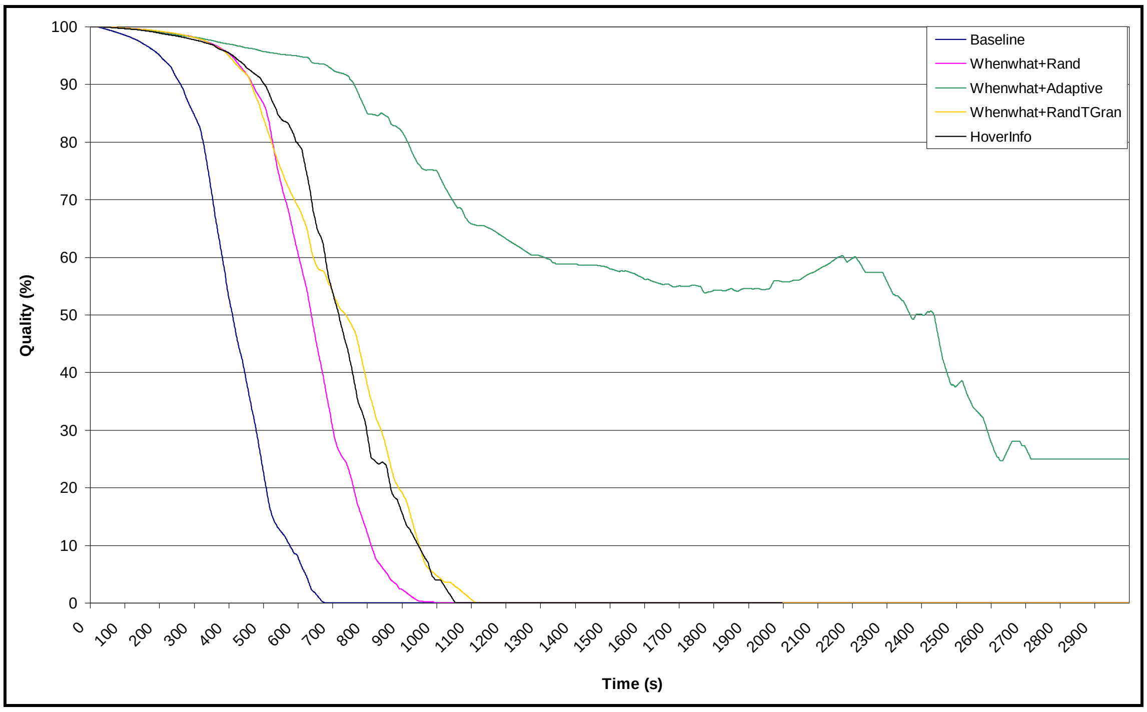


Figure 5.9 Temporal data quality at transmission power -5 dBm

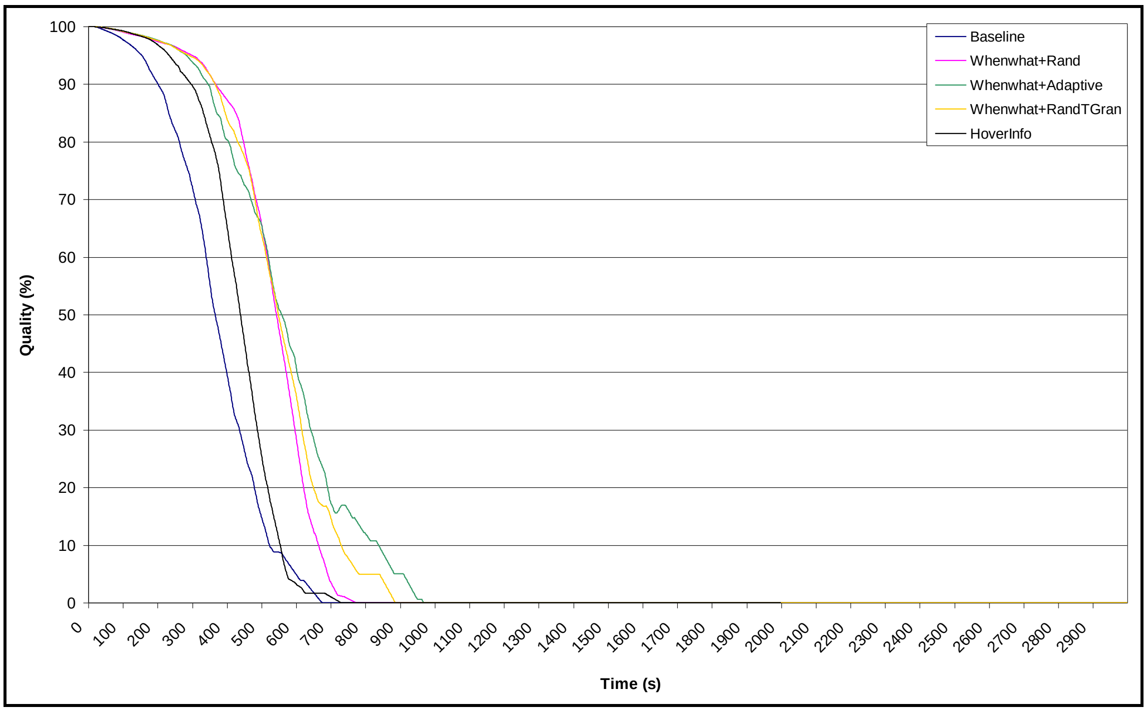


Figure 5.10 Temporal data quality at transmission power -10 dBm

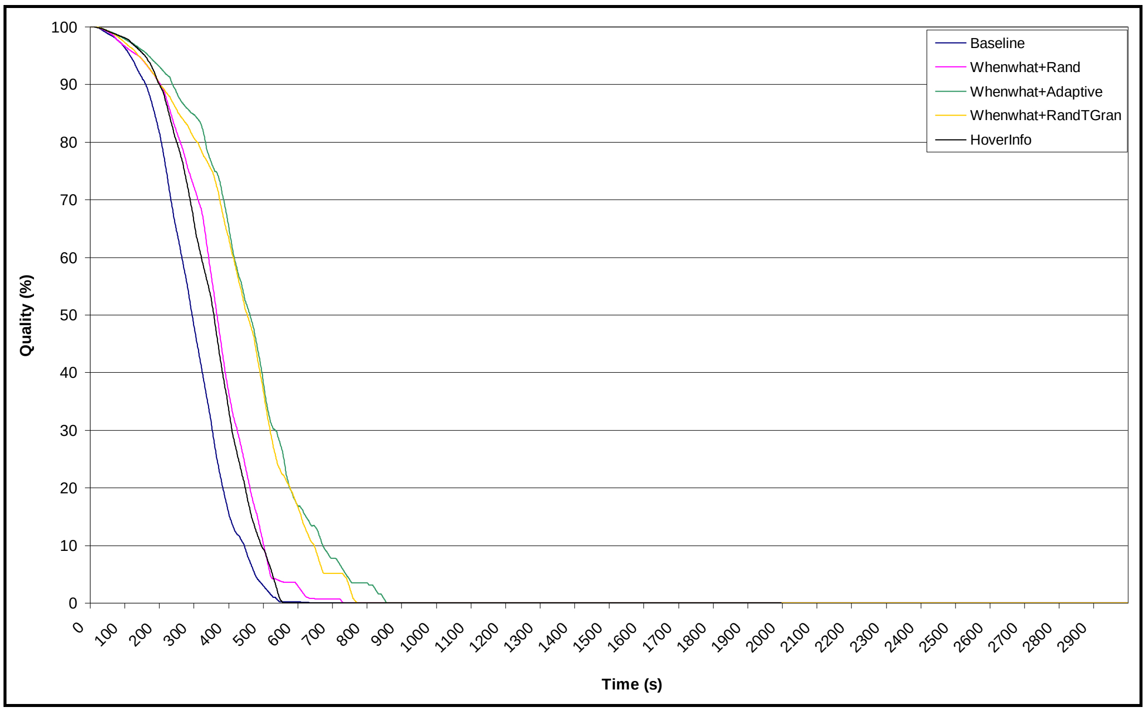


Figure 5.11 Temporal data quality at transmission power -15 dBm

5.4 Data Availability * Temporal Data Quality

The performance of the Data Hovering algorithm can be expressed as the product of the data availability multiplies the data quality. Figure 5.12, Figure 5.13, Figure 5.14 and Figure 5.15 show the data availability multiplies the temporal data quality with different transmission powers. These figures show that the Random approach performs better than the Baseline, the RTG approach has the same level of performance as the Random, the Adaptive approach performs better over a long time period, and the difference between them is smaller when reducing the transmission power level.

Comparing with the Figure 5.1 which is illustrating the data availabilities of different policies at 0 dBm, it is lack of difference between Figure 5.1 and Figure 5.12. This is because the temporal data qualities (Figure 5.5) remain at a high level until their availabilities become lower. When the data availability is high, it has higher probability to achieve higher data quality which has already been discussed in section 5.3.1. In this case, the data quality has less impact on the multiplication of the data availability and the data quality. However, it is still necessary to measure the data quality when the data transmission time is limited. With limited data transmission time, only partial data can be transmitted, and the data, which has not been transmitted, will no longer be retained in their area of origin. Thus, it reduces the data availability. In this case, the data quality is important factor to be considered when the data availability is lower, because it indicates whether the limited retained data represent more information.

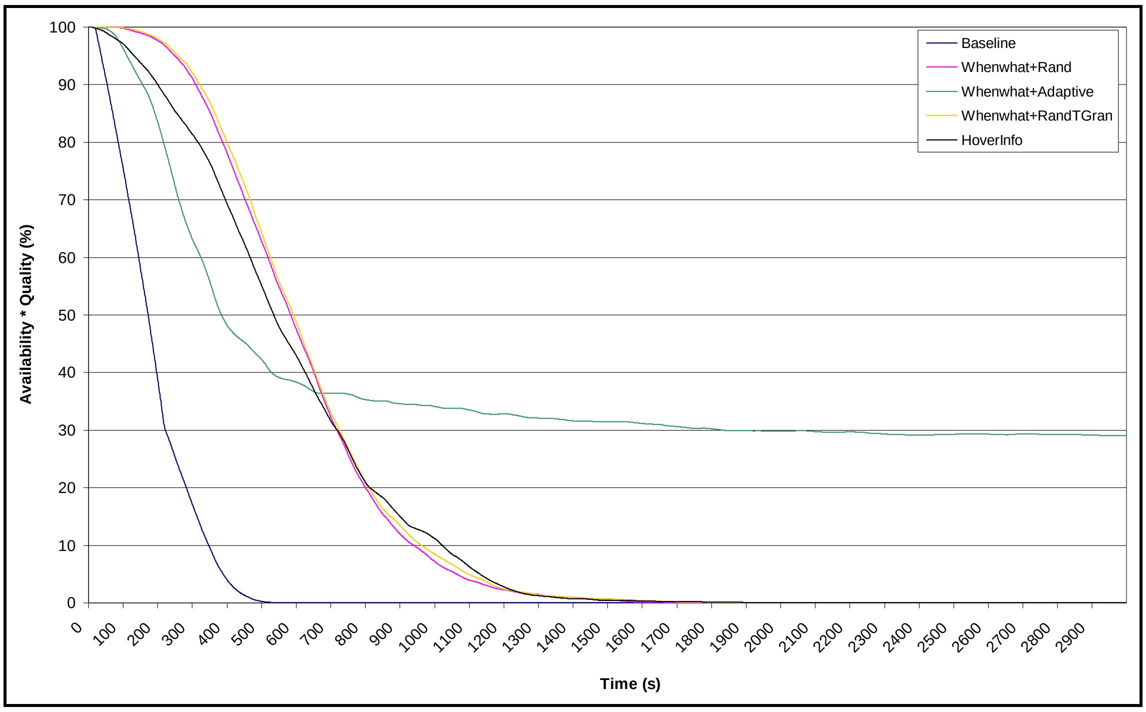


Figure 5.12 Data availability * Temporal data quality at transmission power 0 dBm

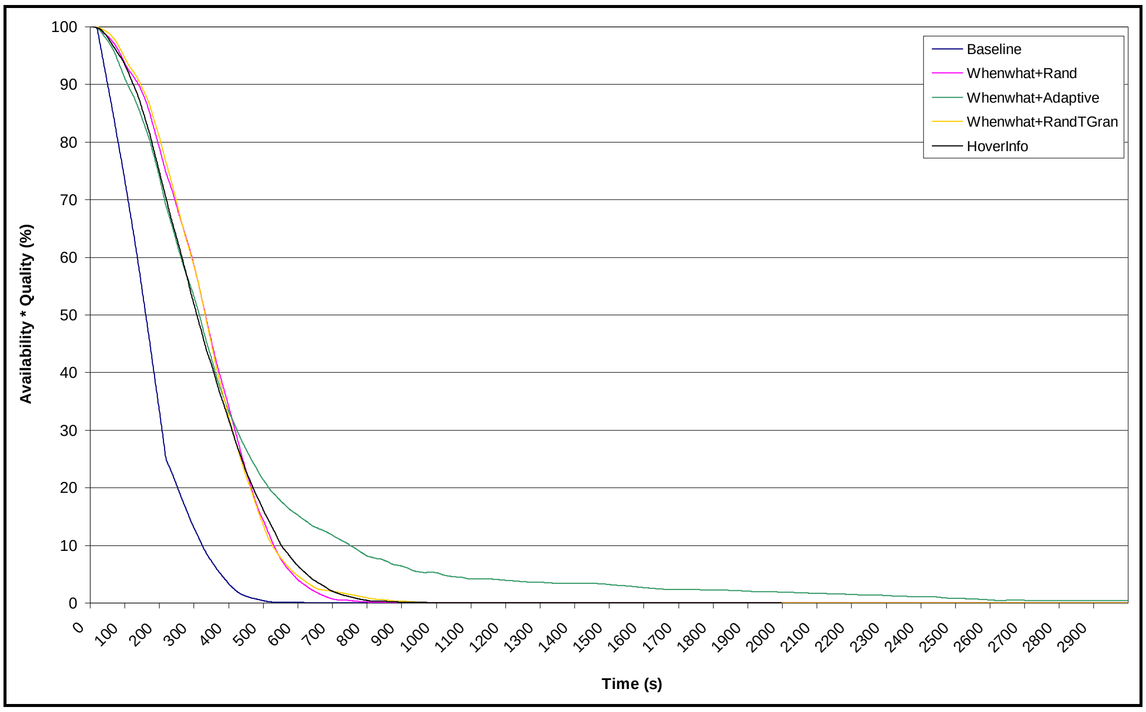


Figure 5.13 Data availability * Temporal data quality at transmission power -5 dBm

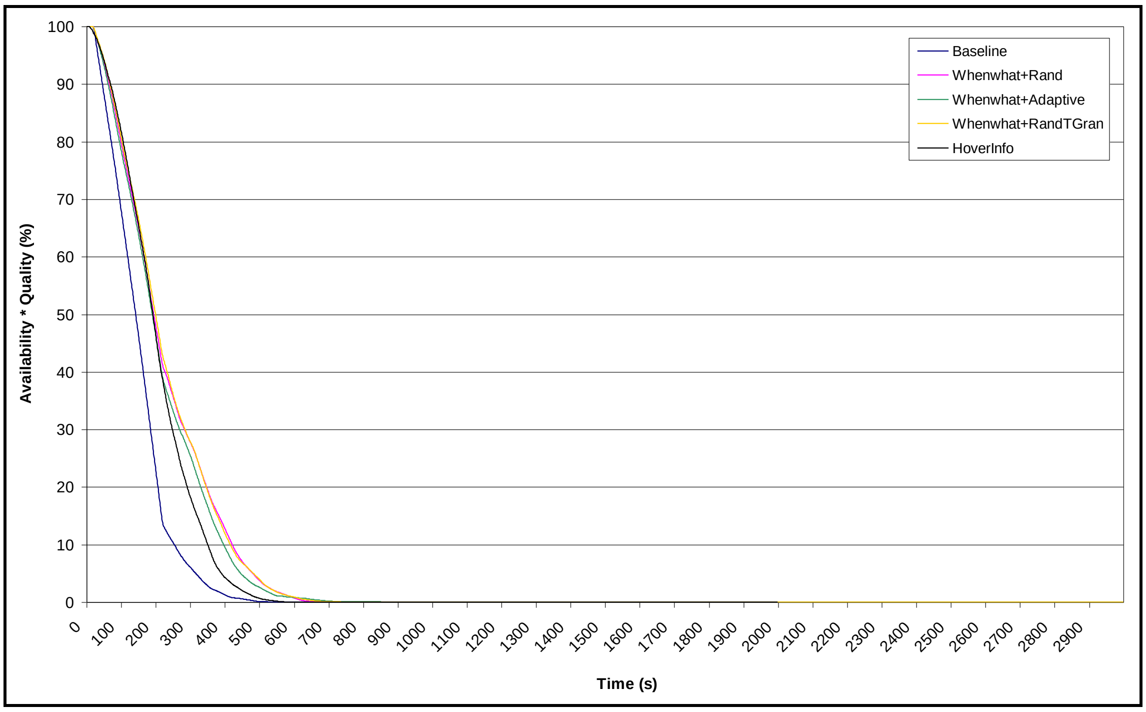


Figure 5.14 Data availability * Temporal data quality at transmission power -10 dBm

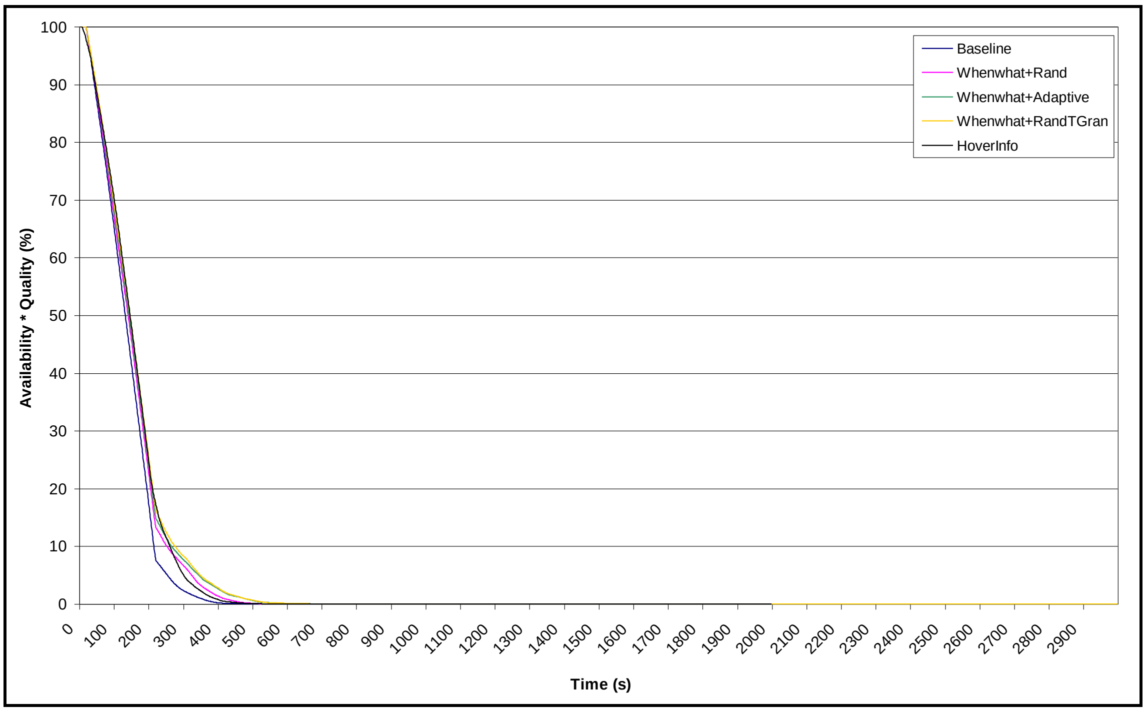


Figure 5.15 Data availability * Temporal data quality at transmission power -15 dBm

5.5 Discussion of Energy-constrained Environment

5.5.1 Overview

Since the network nodes in the Wireless Sensor Network are usually powered by battery, the energy is the scarcest resource. Thus, the power consumption is one of the major considerations for designing Data Hovering algorithm in energy-constraint environment. Less power consumption leads the longer network lifetime. The transmission power level of a network node plays an important role in energy consumption and the signal strength. Higher transmission power level consumes more energy, and it also amplifies the signal strength which means the communication range is greater. In addition, since the increase of the communication range results the increase of number of retained data, so the Data Hovering algorithm performs better when the product of its data availability and its temporal data quality is higher and the transmission power level is fixed. On the other hand, the Data Hovering algorithm will consume less energy to achieve the same performance as the others as its transmission power level is lower.

5.5.2 Comparison of Random and Baseline

Figure 5.16, Figure 5.17 and Figure 5.18 illustrate the product of the data availability and temporal data quality for Baseline which its transmission power is 0dBm and Random which its transmission powers are -10dBm and -15dBm, the performance of the Baseline transmitting data at -5dBm and the Random transmitting data at -10dBm and -15dBm, and the Baseline transmitting at -10dBm and the Random transmitting at -15dBm, respectively. These figures suggest that the Random approach consumed less energy than the Baseline to achieve the same performance, since the transmission power level of the Random is lower than the Baseline.

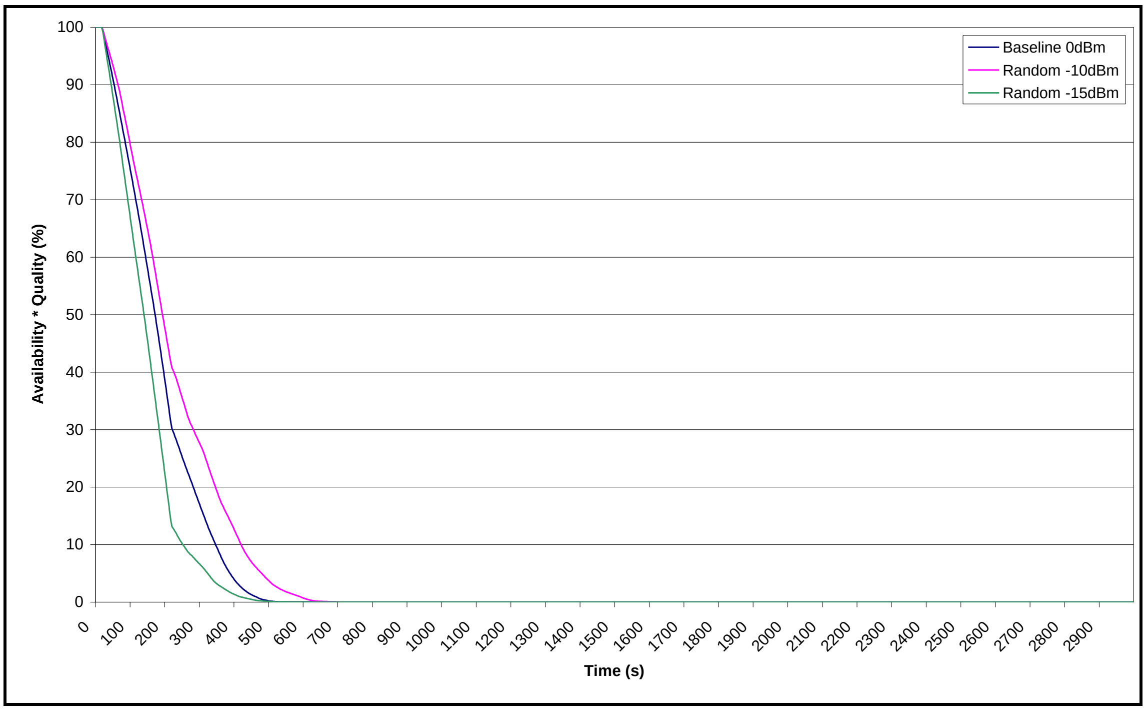


Figure 5.16 Data availability * Temporal data quality for Baseline at 0dBm, Random at -10dBm, and Random at -15dBm

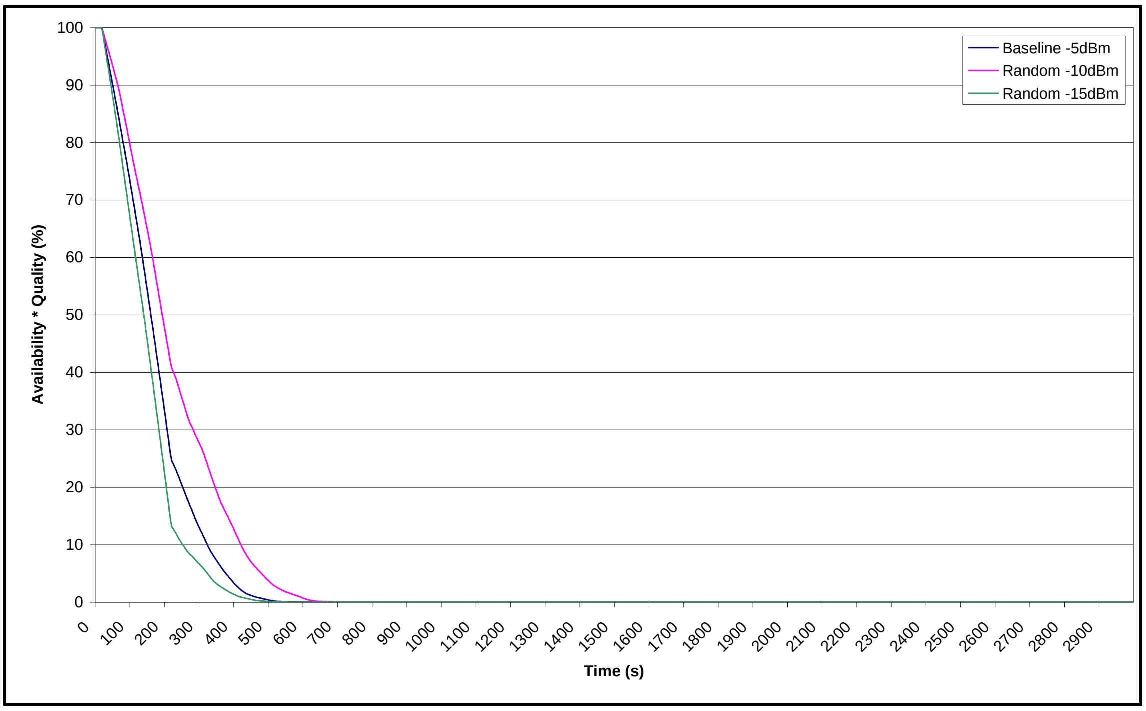


Figure 5.17 Data availability * Temporal data quality for Baseline at -5dBm, Random at -10dBm, and Random at -15dBm

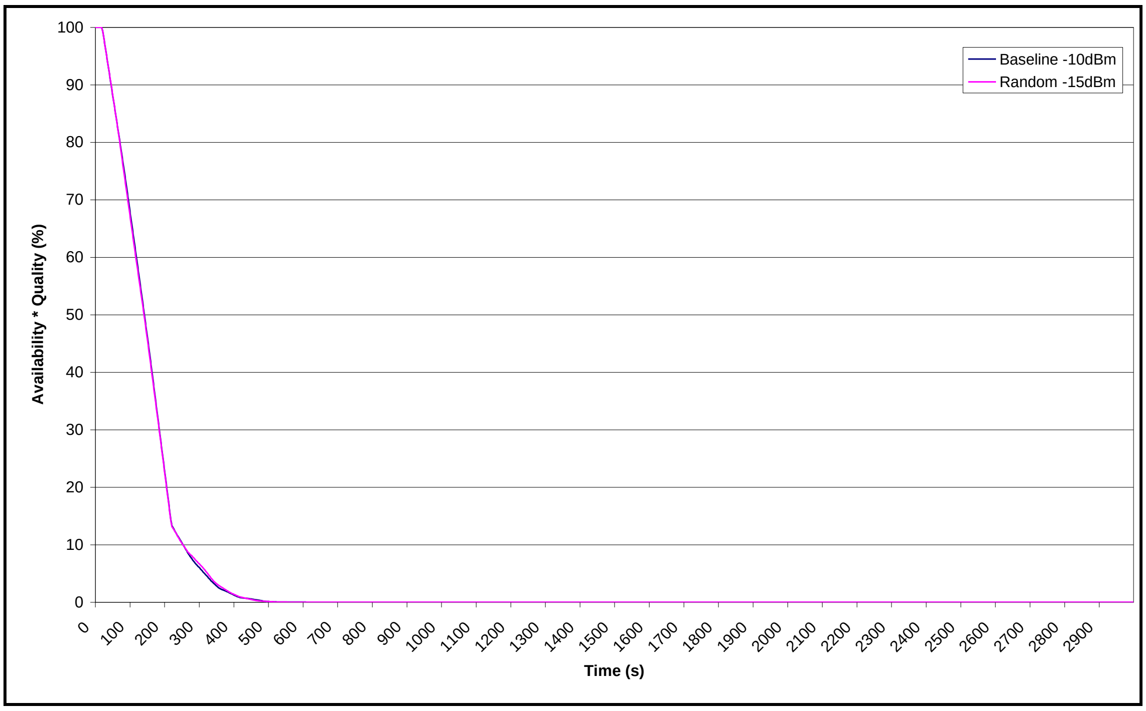


Figure 5.18 Data availability * Temporal data quality for Baseline at -10dBm, Random at -15dBm

5.5.3 Comparison of RTG and Random

The RTG approach consumes almost the same amount energy as the Random approach to achieve the same performance, since their performances are at the same level under the same transmission power level as discussed in section 5.4.

5.5.4 Comparison of Adaptive and Baseline

Figure 5.19 illustrates the product of the data availability and temporal data quality for Baseline which its transmission power is 0dBm and Adaptive which its transmission powers are -10dBm and -15dBm. The performance of the Baseline is worse than the Adaptive at -10dBm but it is better than the Adaptive at -15dBm. In order to achieve the same performance as the Baseline at 0dBm, the transmission power of the Adaptive should be between -10dBm and -15dBm. Figure 5.20 suggests that the transmission power level of Adaptive is also between -10dBm and -15dBm to achieve the same performance as the Baseline transmitting data at -5dBm, and the Adaptive requires -15dBm for transmitting data in order to achieve the same performance as the Baseline transmitting data at -10dBm which has been shown in Figure 5.21. Therefore, the energy consumption of the Adaptive approach is less than the Baseline for achieving the same performance, because Adaptive requires lower transmission power level.

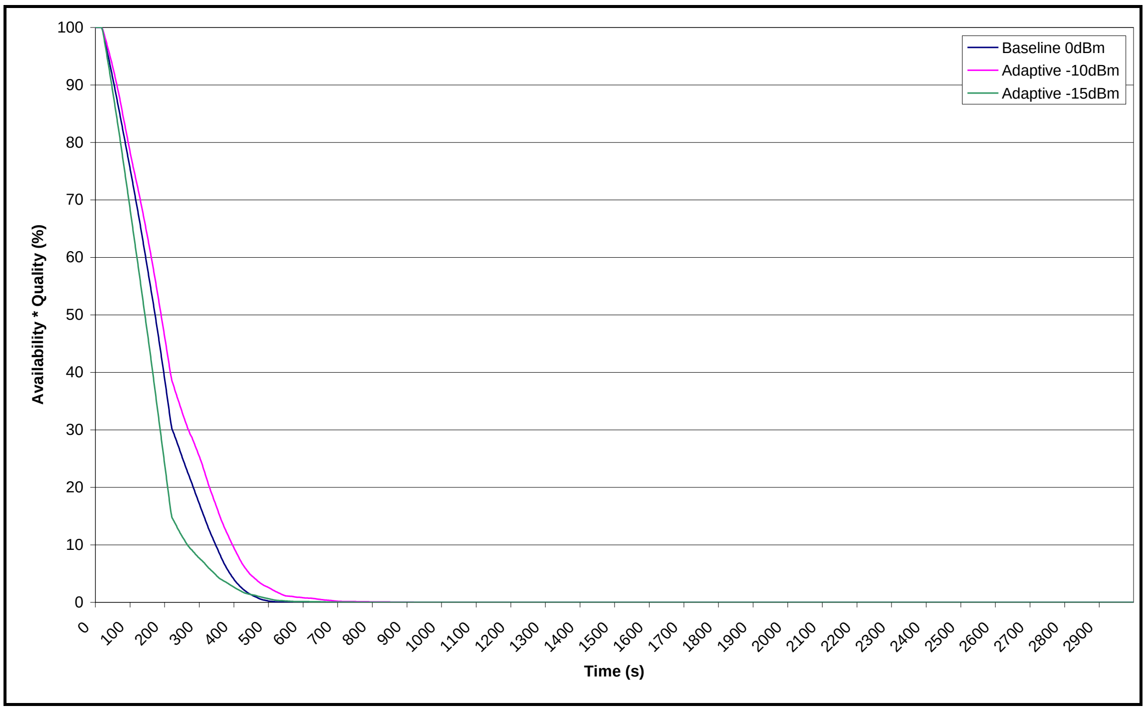


Figure 5.19 Data availability * Temporal data quality for Baseline at 0dBm, Adaptive at -10dBm, and Adaptive at -15dBm

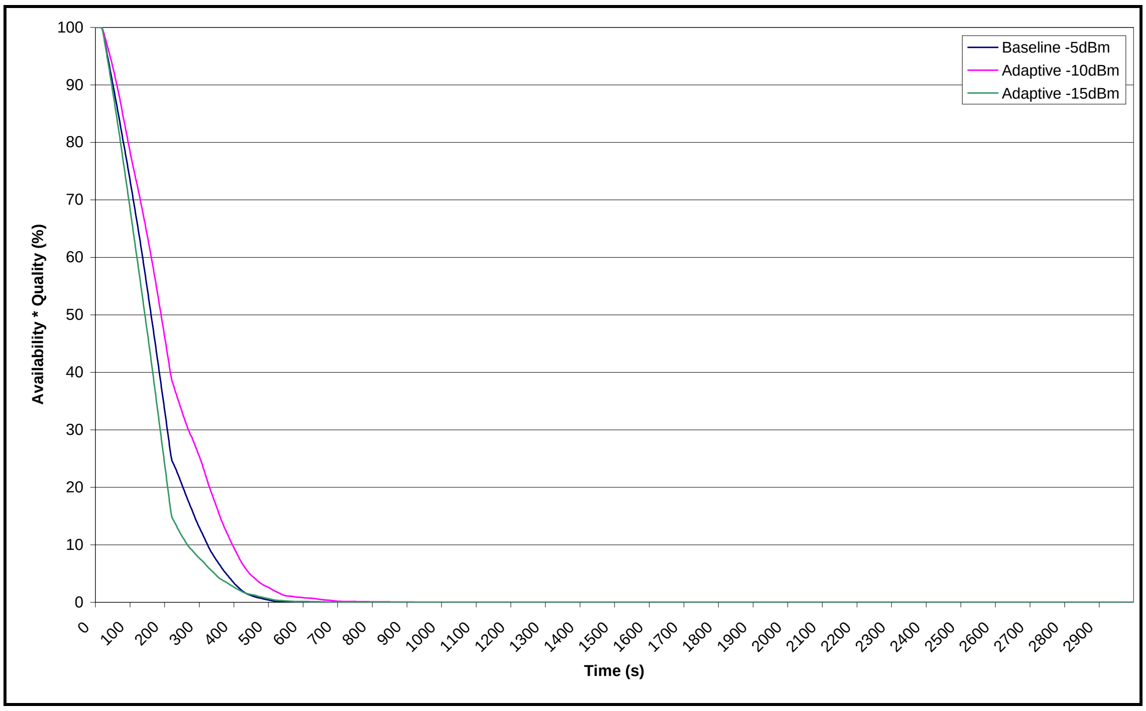


Figure 5.20 Data availability * Temporal data quality for Baseline at -5dBm, Adaptive at -10dBm, and Adaptive at -15dBm

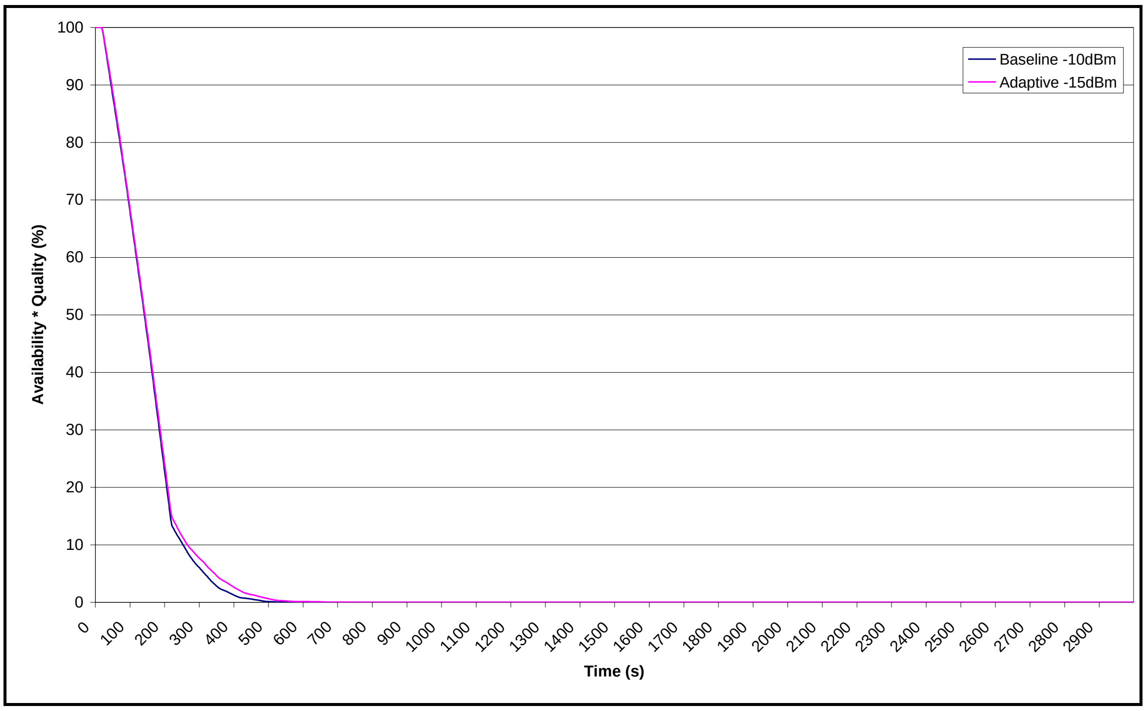


Figure 5.21 Data availability * Temporal data quality for Baseline at -10dBm, Adaptive at -15dBm

5.5.5 Comparison of Adaptive and Random

It is not possible to compare the energy consumptions of Random and Adaptive algorithms at higher transmission power level (e.g. 0dBm and -5dBm) by determining how much transmission power should be used by one particular algorithm for achieving the same performance as another one. This is because, at higher transmission power level, the Adaptive outperforms over a long time period, but the Random performs better during the early stage. As shown in Figure 5.22, the performance of the Adaptive transmitting at 0dBm was the almost same as the Random transmitting at -5dBm before 300 seconds, and it is significantly higher than the Random afterwards. When comparing the performance of the Adaptive transmitting at 0dBm with the Random transmitting at 0dBm, and the Adaptive transmitting at -5dBm and the Random transmitting at -5dBm which has been illustrated in Figure 5.23, it has been observed that the Adaptive performs worse than the Random during the early time, and then it outperforms the Random.

At lower transmission power level, the performance of these two algorithms are almost the same when both transmitting at the same power level. In Figure 5.24, the curves showing the performances of the Adaptive and the Random which both were transmitting data at -10dBm are almost overlapped. Thus, the transmission power level of these two algorithms will be the same for achieving the performance when both transmission powers are low, and hence their energy consumptions will be the same at lower transmission power level.

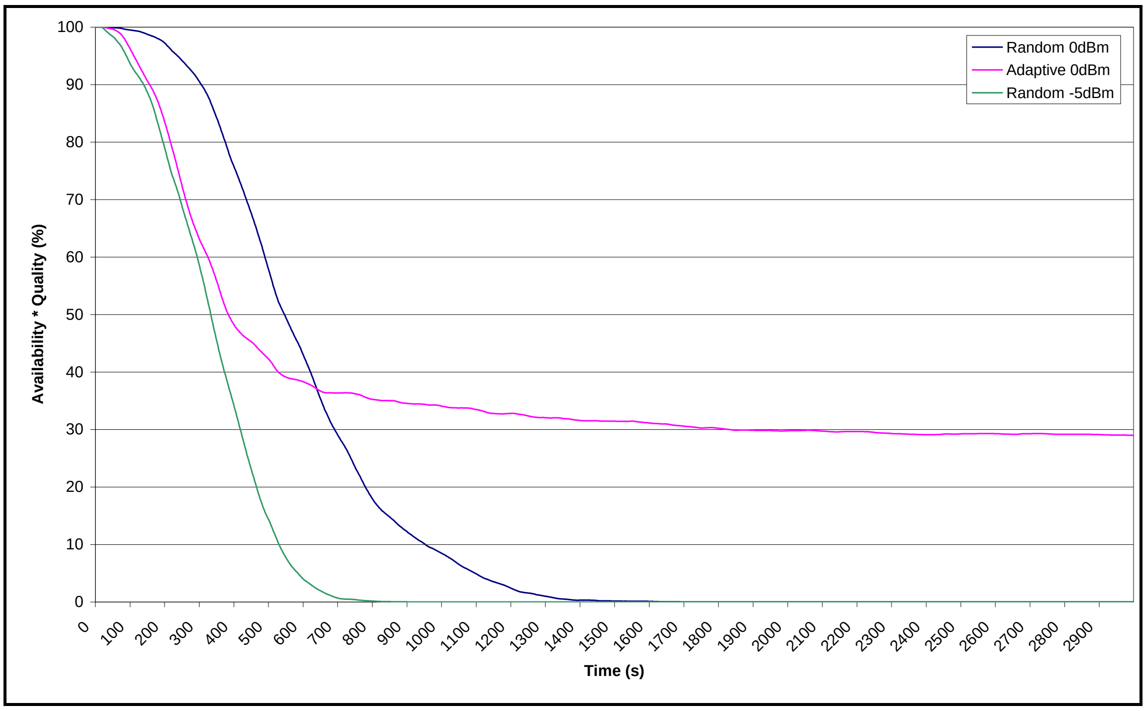


Figure 5.22 Data availability * Temporal data quality for Random at 0dBm, Random at -5dBm, and Adaptive at 0dBm

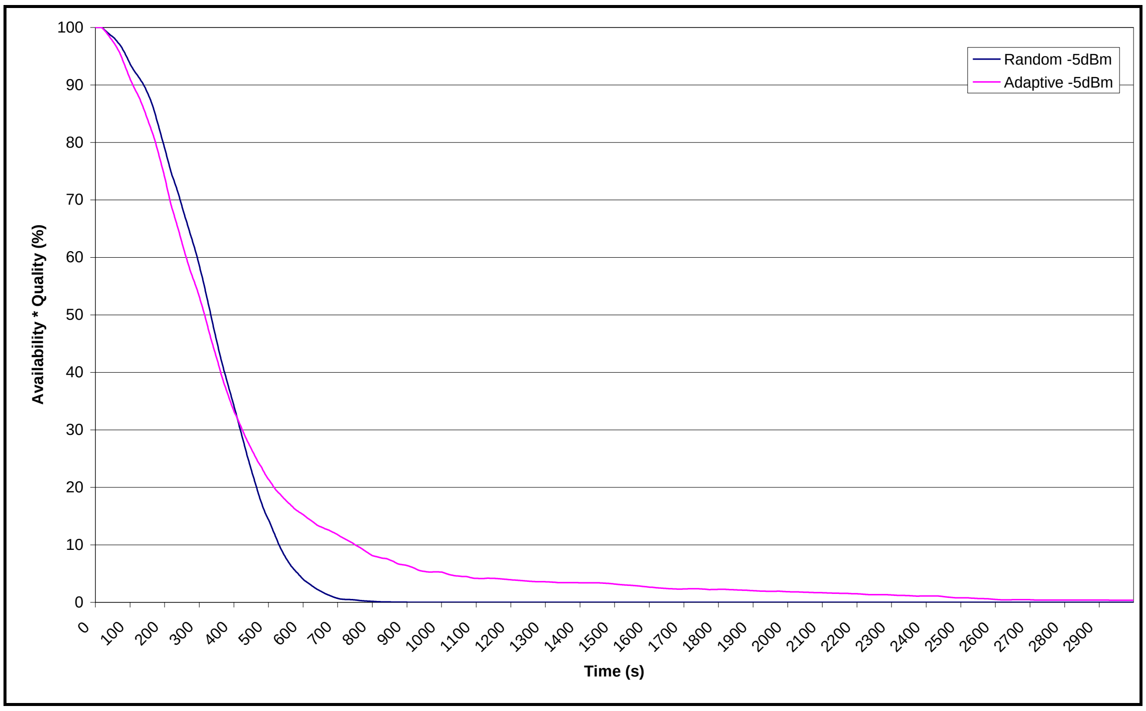


Figure 5.23 Data availability * Temporal data quality for Random at -5dBm and Adaptive at -5dBm

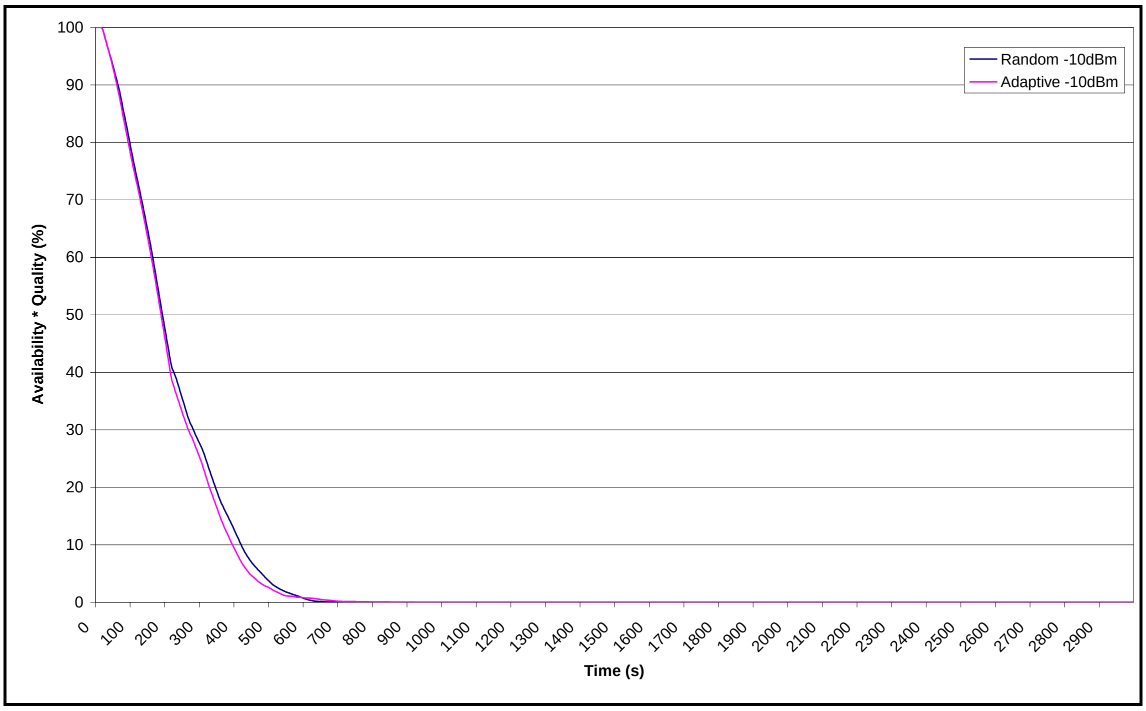


Figure 5.24 Data availability * Temporal data quality for Random at -10dBm and Adaptive at -10dBm

5.6 Summary

This chapter evaluated the proposed Data Hovering algorithms with different defined policies by comparing their simulation results of data availability and temporal data quality, and followed by a discussion of how these algorithms perform in an energy-constrained environment by comparing their transmission power levels to achieve the same performance.

In terms of data availability, the defined Data Hovering algorithms complying with "when to transmit" and "what to transmit" policies outperform the Baseline. This is because "when to transmit" and "what to transmit policies" ensure the node only transmit the relevant data instead of transmitting all the data in its memory when the number of data can be transmitted is limited. The RTG has almost the same performance as the Random where they have the same data availability during the early stage of the simulation, the Random performs a little bit better after a while and RTG outperforms at the later stage. Comparing with the Random and RTG, the Adaptive has the lowest data availability at the early stage, and it outperforms over the longer simulation time. This is because the Random, RTG and Adaptive have the same "when to transmit" and "what to transmit" policies but with the different prioritization schemes. By having the same "when to transmit" and "what to transmit" policies, the algorithm performs better if the probability of transmitting same data by different nodes is lower. However, the more duplicate data transmitted results these transmitted data will have the higher survival rate. When the total number of data related to the network decreases, these data would always be transmitted by different nodes. Thus, the data availability will then be decreased slower. The difference between the data availabilities of different algorithms becomes smaller when the transmission power level of the node decreases.

In terms of temporal data quality, the Random outperforms the Baseline. This is because when the data transmitting order of both algorithms are random, higher data availability leads to higher probability to have higher data quality (more details can be found in section 5.3.1). By having the same "when to transmit" and "what to transmit" policies, the data quality will be higher if the retained data are more spread the sampling time period. The RTG has almost the same temporal data quality as the Random at the early stage and it outperforms over the longer time period. This is because the RTG outperforms the Random when the number of transmitted data of each individual node is less or equal to the number of its time segments, and their difference becomes smaller when more data will be transmitted by each node (detailed explanation can be found in section 5.3.2). The Adaptive outperforms all the other algorithms. The difference between the temporal data qualities of different algorithms becomes smaller when the transmission power level of the node decreases.

In an energy-constrained environment, lower transmission power level leads to lower energy consumption. An algorithm would outperform another if it can achieve the same performance, which is the product of data availability and the data quality, with lower transmission power level. Through the comparison, the Random, RTG and Adaptive outperform the Baseline. The RTG will consume the same amount of energy as the Random to achieve the same performance. In addition, it is not possible to compare the energy consumption of the Random and the Adaptive at higher transmission power level. However, they consume the same amount of energy to achieve the same performance at lower transmission power level.

Chapter 6

Conclusions and Future Work

This chapter concludes this thesis with summary of the achievements and highlights the suggestions of potential future work.

6.1 Conclusions

This thesis addressed the issue of sensed data getting lost from its area of origin in a Mobile Wireless Sensor Network (WSN). Architecture of Data Hovering for improving the data availability as well as the temporal quality of the retained data has been defined. A family of Data Hovering policies have been defined and implemented in a network simulator. The performances of the defined Data Hovering algorithms have also been evaluated in this thesis.

The problem of Data Hovering is caused by movement of the network nodes. The location-based data will move out of its area of origin with its carrying node. Based on the concept of Data Hovering, the particular policies of Data Hovering which need to be defined have been investigated. In addition, due to the unique characteristics of WSN, some constraints of WSN arise. These constraints must be taken into consideration when defining the Data Hovering policies.

The investigation of existing approaches in various fields which are related to Data Hovering was then carried out. To explore the limitations and research gap in Data Hovering, the approaches with the aim to replicate the data in data's attached area and the approaches with the aim to retain the data in its area of origin have been examined by investigating how their proposed algorithm defined the Data Hovering policies. All these approaches were proposed in either Mobile Ad Hoc Networks (MANET) or Vehicular Ad Hoc Networks (VANET). Due to the differences among the characteristics of WSN, MANET and VANET, the limitations of the existing approaches (described in section 2.3) arise which requiring the Data Hovering policies need to be redefined by considering the unique characteristics of WSN.

The complete Data Hovering algorithms complying with different policies have been developed, in order to improve the data availability and temporal data quality. In particular, the defined “when to transmit” policy ensures the nodes start and stop transmission at the appropriate time, the “what to transmit” policy ensures only the appropriate data will be transmitted when the transmission triggered, and the data prioritization policy attempts retain numerous data which can represent the different information.

These defined policies have been implemented and simulated in OMNeT++ simulator with Castalia extension with the specific parameters settings. To evaluate the performances of the proposed Data Hovering algorithms, the evaluation metrics consisting of data availability and temporal data quality have been defined.

A baseline complying with the simple policies has been determined in order to compare the performance with the defined Data Hovering algorithms. Through the analysis of the experimental results, it has been observed that the Data Hovering algorithm complying with defined policies outperform the baseline in terms of data availability. The data availabilities for the random temporal granularity prioritization (RTG) and the random prioritization (Random) are almost the same with minor variances. Furthermore, at higher transmission power level, the algorithm with random prioritization performs better during the early time of the simulation, and the algorithm with adaptive temporal granularity prioritization (Adaptive) outperforms over a longer time period. The difference between the data availabilities becomes smaller when the transmission power level decreases. In terms of temporal data quality, all the proposed algorithms outperform the baseline at higher transmission power level. The temporal data quality of Adaptive remains at a very high level despite its data availability has already decreased. The RTG outperforms the Random when the number of data will be transmitted by each individual node is less or equal to the number of its time segments. The difference between the temporal data qualities of these algorithms with different policies becomes smaller if the transmission power level is lower. In energy constrained environment, less transmission power level consumes less energy. The proposed policies can achieve the same performance as the Baseline, which is the product of data availability and temporal data quality, by using lower transmission power level. Thus, the Data Hovering algorithm with these policies consumes less energy in order to achieve the same performance as the baseline. The RTG consumes the same amount of energy to achieve the same performance as the Random. Moreover, it is not possible to compare the energy consumption of the Random and the Adaptive at higher transmission power level. This is because the Adaptive performs better in a longer time, but the Random performs better during the early time. However, their performances are at the same level under lower transmission power level, so that they are likely to consume the same level of energy to achieve the same performance when the transmission power is low.

6.2 Future Work

6.2.1 Adaptive Spatial Granularity Prioritization

This thesis has defined the adaptive temporal granularity prioritization policy, in order to spread the retained data over their sampling time period. This prioritization policy ensures that the retained data would represent more different information of their area of origin. Another alternative approach for the same purpose is to spatially spread the retained data in their area of origins. Thus, data from different sub areas of the network Grids would be retained. The performance of the adaptive prioritization has been analyzed through comparing the experimental results with the baseline and the proposed Data Hovering algorithm with other prioritization policies. It shows that this prioritization policy outperforms the baseline, and outperforms other prioritization policies over a long time period in terms of data availability. Its temporal data quality is much higher than other policies. Since the main technique for defining the spatial prioritization policy is the same as temporal prioritization, the performance of the spatial prioritization should be the same as the temporal prioritization. However, comparing with the temporal prioritization, the spatial prioritization requires to be defined in a two dimensional area rather than a one dimension of sampling time period. Thus, the temporal prioritization needs to be adapted for converting to spatial prioritization. Moreover, the evaluation metric for calculating the spatial quality of the retained data must also be defined. In addition, further experiments would be carried out to verify the performance of spatial prioritization.

6.2.2 Limited Memory

Although the proposed Data Hovering algorithms assume the network nodes have unlimited memory, the experiments include a concept of Garbage Data. By means of this concept, the data will be set to Garbage when its carrier is reaching the boundaries of the network area, so that these data are no longer related to any Grids in the network. Since the total number of data is constant in the experiments, this concept aims to prevent all the nodes carrying all the initial data in a long simulation run. If this happens, then it is not necessary to transmit data to other nodes in order to retain the data in its area of origin. This is because all the data will be available when there is at least one node in the area of origin of the data. The concept of the Garbage data is similar to limited memory which the data will be removed from the memory of their carrying node. However, it is still necessary to define a complete cleansing policy which including "what to receive", "when to delete" and "what to delete" policies for real world applications with limited memory, because the total number of data would still be increased when nodes are taking samples.

6.2.3 Dynamic sensed data creation

In the defined Data Hovering algorithms, the sensing ability of the nodes has been disabled. Instead, a certain number of data are initially carried by network nodes. Both the defined adaptive temporal granularity prioritization and the evaluation metrics are based on this assumption. However, data are being collected by nodes in a certain time interval in real world applications. Thus, it is necessary to redefine the adaptive temporal granularity prioritization policy to involve the new created sensed data. The possible adaptation to this prioritization would be constructing the time segment indices based on current existing data of a Grid_Data of their carrying node, when the start transmission triggered. The existing data will then be prioritized by reordering their positions in the memory of their carrier, after computing their individual time segment index. Moreover, the calculation of the evaluation metrics needs to be redefined by considering dynamical total number of sensed data in the network.

6.2.4 Collaborative Data Hovering

The proposed Data Hovering algorithms in this thesis are autonomous, in which the nodes decide which data should be transmitted depending on their own locations and the gathering locations of their carrying data, but without requiring the knowledge of their neighbouring nodes. Considering some nodes are located in a sub area of a particular Grid and the size of this Grid can be covered by the communication range of these nodes, a piece of data related to this Grid is available if it exists in the memory of at least one of these nodes, so that it can be accessed by other nodes which are joining this Grid. However, this may waste the memory of the nodes if there is more than one copy of the same data storing in different nodes. It is worth to define a collaborative approach which each node decides which data to be transmitted and which data should be received based on the knowledge of its neighbouring nodes in order to address this issue. Furthermore, a hybrid wine and milk [47, 50] architecture could also be defined to allow different nodes to keep different data which were collected at different times or areas. This would lead to the less consumption in bandwidth, because the data availability would be the same as autonomous approach but the total number of data requiring to be transmitted is less.

6.2.5 Hybrid Pull-Push approach

In “what to transmit” policy of the proposed Data Hovering algorithm, the data related to the previous leaving Grid or previous of the previous leaving Grid of its carrying node will be transmitted. Under some situations, it is possible that a node which is joining a Grid did not receive any data related to the new Grid. For instance, no nodes were moving out of such Grid, or this node was not in the communication range of any nodes which were transmitting the data related to such Grid. In addition, it is possible that nodes within the communication range of this new joining node may not have the data related to this current Grid. In this case, this node may not be able to transmit the enough data of this current Grid when it is leaving, so that reducing the data availability. To resolve this problem, a hybrid pull and push approach could be designed. In this approach, the data will be transmitted when its carrier leaving its area of origin, so the data is pushed to the neighbouring nodes of its carrier. In addition, when a node joining a Grid and it finds the relevant data within its communication range is lower than a certain threshold, it periodically requests the relevant data from its neighbouring nodes, so pulling the data.

References

[1] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "A Survey on Sensor Networks," *Communications Magazine, IEEE,* vol. 40, pp. 102 - 114, 2002.

[2] I. Khemapech, I. Duncan, and A. Miller, "A Survey of Wireless Sensor Networks technology," in *Proc. of The 6th Annual Postgraduate Symposium on The Convergence of Telecommunications, Networking and Broadcasting*, 2005.

[3] P. Rulikowski, R. Martinez, and J. Barrett, "Sensor Network Hardware Infrastructure for Smart Spaces," in *1st International Workshop on Managing Ubiquitous Communications and Services (MUCS),* 2003.

[4] J. Feng, F. Koushanfar, and M. Potkonjak, "System-architectures for Sensor Networks Issues, Alternatives, and Directions," in *Computer Design: VLSI in Computers and Processors, 2002. Proceedings: 2002 IEEE International Conference on*, 2002, pp. 226-231.

[5] M. A. M. Vieira, C. N. Coelho, Jr., D. C. da Silva, Jr., and J. M. da Mata, "Survey on Wireless Sensor Network Devices," in *Emerging Technologies and Factory Automation, 2003. Proceedings: ETFA '03. IEEE Conference*, 2003, pp. 537-544 vol.1.

[6] Y. Yu, *Information Processing and Routing in Wireless Sensor Networks*: World Scientific Publishing Co., Inc. , 2007.

[7] A. Sinha and A. Chandrakasan, "Dynamic Power Management in Wireless Sensor Networks," *Design & Test of Computers, IEEE,* vol. 18, pp. 62-74, 2001.

[8] P. Padmanabhan and G. S. Kang, "Real-time Dynamic Voltage Scaling for Low-power Embedded Operating Systems," vol. 35, ACM, 2001, pp. 89-102.

[9] *Pico Radio*. Available: http://bwrc.eecs.berkeley.edu/research/pico_radio/

[10] *The Sensor Network Museum*. Available: http://www.snm.ethz.ch/Main/HomePage

[11] *MICAz Datasheet*. Available: http://www.openautomation.net/uploadsproductos/micaz_datasheet.pdf

[12] P. Joseph, S. Robert, and C. David, "Telos: Enabling Ultra-low Power Wireless Research," presented at the Proceedings of the 4th international symposium on Information processing in sensor networks, Los Angeles, California, 2005.

[13] *Tmote Sky Datasheet*. Available: http://www.eecs.harvard.edu/~konrad/projects/shimmer/references/tmote-sky-datasheet.pdf

[14] S. Roundy, P. K. Wright, and J. Rabaey, "A Study of Low Level Vibrations as a Power Source for Wireless Sensor Nodes," *Computer Communications*, vol. 26, pp. 1131-1144, 2003.

[15] A. Savvides and M. B. Srivastava, "A Distributed Computation Platform for Wireless Embedded Sensing," in *Computer Design: VLSI in Computers and Processors, 2002. Proceedings: 2002 IEEE International Conference on*, 2002, pp. 220-225.

[16] P. Sung, I. Locher, A. Savvides, M. B. Srivastava, A. Chen, R. Muntz, and S. Yuen, "Design of a Wearable Sensor Badge for Smart Kindergarten," in *Wearable Computers, 2002. (ISWC 2002). Proceedings: Sixth International Symposium on*, 2002, pp. 231-238.

[17] *BTnodes - A Distributed Environment for Prototyping Ad Hoc Networks*. Available: www.btnode.ethz.ch

[18] S. Gyula, Mikl, M. s, ti, L. kos, deczi, Gy, B. rgy, K. Branislav, Andr, N. s, das, P. bor, S. nos, and F. Ken, "Sensor Network-based Countersniper System," in *Proceedings of the 2nd international conference on Embedded networked sensor systems*, Baltimore, MD, USA, 2004.

[19] Y. Liyang, W. Neng, and M. Xiaoqiao, "Real-time Forest Fire Detection with Wireless Sensor Networks," in *Wireless Communications, Networking and Mobile Computing, 2005. Proceedings: 2005 International Conference on*, 2005, pp. 1214-1217.

[20] M. V. Ramesh, "Real-Time Wireless Sensor Network for Landslide Detection," in *Sensor Technologies and Applications, 2009. SENSORCOMM '09. Third International Conference on*, 2009, pp. 405-409.

[21] K. Selvarajah, A. Tully, and P. T. Blythe, "Integrating Smartdust into the Embedded Middleware in Mobility Application (EMMA) Project," *Intelligent Environments, 2008 IET 4th International Conference on,* pp. 1-8, 2008.

[22] C. R. Baker, K. Armijo, S. Belka, M. Benhabib, V. Bhargava, N. Burkhart, A. Der Minassians, G. Dervisoglu, L. Gutnik, M. B. Haick, C. Ho, M. Koplow, J. Mangold, S. Robinson, M. Rosa, M. Schwartz, C. Sims, H. Stoffregen, A. Waterbury, E. S. Leland, T. Pering, and P. K. Wright, "Wireless Sensor Networks for Home Health Care," in *Advanced Information Networking and Applications Workshops, 2007, AINAW '07. 21st International Conference on*, 2007, pp. 832-837.

[23] Mark L. McKelvin, Jr., L. W. Mitchel, and M. B. Nina, "Integrated Radio Frequency Identification and Wireless Sensor Network Architecture for Automated Inventory Management and Tracking Applications," in *Proceedings of the 2005 conference on Diversity in computing*, Albuquerque, New Mexico, USA, 2005.

[24] I. Vasilescu, K. Kotay, D. Rus, M. Dunbabin, and P. Corke, "Data Collection, Storage, and Retrieval with an Underwater Sensor Network," presented at the Proceedings of the 3rd international conference on Embedded networked sensor systems, San Diego, California, USA, 2005.

[25] L. Mo and L. Yunhao, "Underground Coal Mine Monitoring with Wireless Sensor Networks," *ACM Transactions on Sensor Networks (TOSN),* vol. 5, pp. 1-29, 2009.

[26] H. Jyh-How, A. Saqib, and M. Shivakant, "CenWits: a Sensor-based Loosely Coupled Search and Rescue System using Witnesses," presented at the Proceedings of the 3rd international conference on Embedded networked sensor systems, San Diego, California, USA, 2005.

[27] Z. Pei, M. S. Christopher, A. L. Stephen, and M. Margaret, "Hardware design experiences in ZebraNet," presented at the Proceedings of the 2nd international

conference on Embedded networked sensor systems, Baltimore, MD, USA, 2004.

[28] A. Villalba and D. Konstantas, "Towards Hovering Information," *Proceedings of the First European Conference on Smart Sensing and Context (EuroSSC 2006),* pp. 161-166, 2006.

[29] H. Labiod, *Wireless Ad Hoc and Sensor Networks*: ISTE Ltd and John Wiley & Sons, Inc, 2008.

[30] G. J. Pottie and W. J. Kaiser, "Wireless Integrated Network Sensors," *Communications of the ACM,* vol. 43, pp. 51-58, 2000.

[31] S. Madden and M. J. Franklin, "Fjording the Stream: An Architecture for Queries over Streaming Sensor Data," *Data Engineering, 2002. Proceedings. 18th International Conference on* pp. 555-566, 2002.

[32] S. Madden, M. J. Franklin, J. M. Hellerstein, and W. Hong, "Tag: A Tiny Aggregation Service for Ad-hoc Sensor Networks," *ACM SIGOPS Operating Systems Review,* vol. 36, pp. 131-146, 2002.

[33] Y. Yao and J. Gehrke, "The Cougar Approach to In-network Query Processing in Sensor Networks " *ACM SIGMOD Rec.,* vol. 31, pp. 9-18, 2002.

[34] H. Yingjie and A. Tully, "Query Processing for Mobile Wireless Sensor Networks: State-of-the-art and Research Challenges," *Wireless Pervasive Computing, 2008. ISWPC 2008. 3rd International Symposium on,* pp. 518-523, 2008.

[35] S. R. Madden, M. J. Franklin, J. M. Hellerstein, and W. Hong, "TinyDB: An Acquisitional Query Processing System for Sensor Networks," *ACM Trans. Database Syst.,* vol. 30, pp. 122--173, 2005.

[36] H. Huang, J. H. Hartman, and T. N. Hurst, "Efficient and Robust Query Processing for Mobile Wireless Sensor Networks," *Global Telecommunications Conference, 2006. GLOBECOM '06. IEEE,* 2006.

[37] Y. Zhang, B. Hull, I. Balakrishnan, and S. Madden, "ICEDB: Intermittently-connected Continuous Query Processing," *Data Engineering, 2007. ICDE 2007. IEEE 23rd International Conference on,* pp. 166-175, 2007.

[38] B. Xu, F. Vafaee, and O. Wolfson, "In-network Query Processing in Mobile P2P databases," *Proceedings of the 17th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pp. 207-216, 2009.

[39] M. Shinohara, T. Hara, and S. Nishio, "Data Replication Considering Power Consumption in Ad Hoc Networks," *Mobile Data Management, 2007 International Conference on*, pp. 118-125, 2008.

[40] D. J. Corbett and D. Cutting, "AD LOC: Collaborative Location-based Annotation," *IPSJ Digital Courier*, vol. 3, pp. 280-292, 2007.

[41] D. J. Corbet and D. Cutting, "Ad loc: Location-based infrastructure-free annotation," *ICMU 2006*, 2006.

[42] I. Leontiadis and C. Mascolo, "Opportunistic Spatio-temporal Dissemination System for Vehicular Networks," *Proceedings of the 1st international MobiSys workshop on Mobile opportunistic networking*, pp. 39-46, 2007.

[43] A. A. V. Castro, G. D. M. Serugendo, and D. Konstantas, "Hovering Information--Self-Organizing Information that Finds its Own Storage," *Autonomic Communication*, pp. 111-145, 2009.

[44] J. L. Fernandez-Marquez, J. L. Arcos, and G. Di Marzo Serugendo, "Infrastructureless Storage in Dynamic Environments," *Proceedings of the 2010 ACM Symposium on Applied Computing*, pp. 1334-1338, 2010.

[45] J. Kangasharju, J. Ott, and O. Karkulahti, "Floating Content: Information Availability in Urban Environments," *Pervasive Computing and Communications Workshops (PERCOM Workshops), 2010 8th IEEE International Conference on*, pp. 804-808, 2010.

[46] A. Xeros, M. Lestas, M. Andreou, A. Pitsillides, and P. Ioannou, "Information Hovering in Vehicular Ad-Hoc Networks," *GLOBECOM Workshops, 2009 IEEE* pp. 1 - 6, 2009.

[47] A. S. Tanenbaum, *Computer Networks*, 4th Edition ed.: Prentice-Hall, 2002.

[48] *OMNeT++ Network Simulation Framework*. Available: www.omnetpp.org/

[49] *Castalia Simulator*. Available: http://castalia.npc.nicta.com.au/

[50] N. Tatbul, "Qos-driven Load Shedding on Data Streams," *XML-Based Data Management and Multimedia Engineering - EDBT 2002 Workshops,* vol. 2490/2002, pp. 779-783, 2002.

Appendix A

Synchronized Node Start Moving

Figure A.1 illustrates the data availabilities of proposed Data Hovering algorithm with random prioritization with (the pink line) and without the start moving delays (the blue line). It shows that, without the start moving delays, the data availability dramatically drops to 88% at approximately 30 seconds, and it returns back to 97% at approximately 150 seconds. This initialization takes place because the nodes are likely to move out their initial located Grid at the same time, so that most of the data suddenly loose from their area of origin. The data availability returns back to a higher level when these data have been transmitted to the nodes which are joining the Grid of these data. To avoid this initialization happening, the node start moving delays, which have been introduced in Section 4.2.6, have been used in the simulation settings.

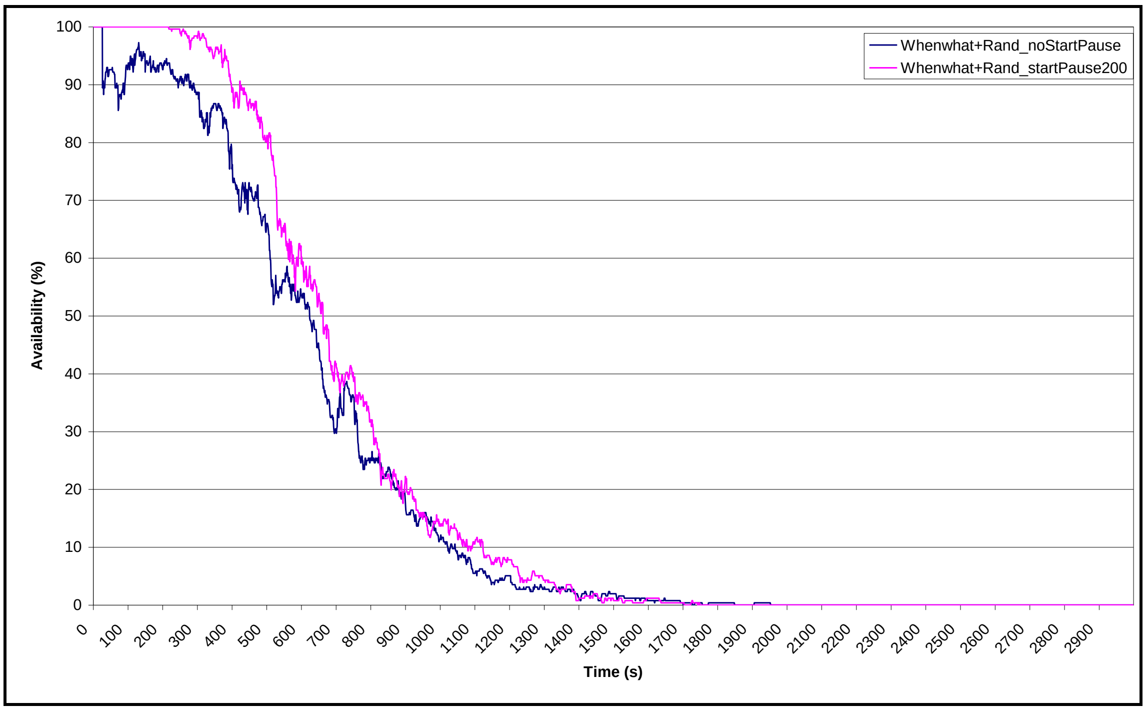


Figure A.1 Data availability of when to transmit + what to transmit + random prioritization with and without start moving delays