Reinforcement Learning Routing Algorithm for Bluetooth Mesh Networks

RASMUS OLSSON
JENS EGELAND
This page intentionally left blank.
Reinforcement Learning Routing Algorithm for Bluetooth Mesh Networks

Rasmus Olsson
Jens Egeland

Supervisor:
Mustafa Özger

Examiner:
Cicek Cavdar
This page intentionally left blank.
Abstract

Today’s office and home environments are moving towards more connected digital infrastructures, meaning there are multiple heterogeneous devices that use short-range communication to stay connected. Mobile phones, tablets, laptops, sensors, printers are examples of devices in such environments. From this, the Internet of Things (IoT) paradigm arises, and to enable it, energy efficient machine-to-machine (M2M) communications are needed. Our study will use Bluetooth Low Energy (BLE) technology for communication between devices, and it demonstrates the impact of routing algorithms in such networks. With the goal to increase the network lifetime, a distributed and dynamic Reinforcement Learning (RL) routing algorithm is proposed. The algorithm is based on a RL technique called Q-learning. Performance analysis is performed in different scenarios comparing the proposed algorithm against two static and centralized reference routing algorithms. The results show that our proposed RL routing algorithm performs better as the node degree of the topology increases. Compared to the reference algorithms the proposed algorithm can handle a higher load on the network with significant performance improvement, due to the dynamic change of routes. The increase in network lifetime with 75 devices is 124% and 100 devices is 349%, because of the ability to change routes as time passes which is emphasized when the node degree increases. For 35, 55 and 75 devices the average node degrees are 2.21, 2.39 and 2.54. On a lower number of devices our RL routing algorithm performs nearly as good as the best reference algorithm, the Energy Aware Routing (EAR) algorithm, with a decrease in network lifetime around 19% on 35 devices and 10% on 55 devices. A decrease in the network lifetime on lower number of devices is because of the cost for learning new paths is higher than the gain from exploring multiple paths.
**Abstract**

Acknowledgements

We would like to thank our supervisor Mustafa Özger for all the guidance, knowledge and support. We would also like to thank our examiner, Cicek Cavdar, for making our project possible.
Contents

1 Introduction 1
   1.1 Background ............................................. 1
   1.2 Problem ................................................. 2
   1.3 Purpose ................................................. 3
   1.4 Goal .................................................. 3
       1.4.1 Benefits, Ethics and Sustainability ............... 3
   1.5 Methodology ............................................. 3
   1.6 Delimitations .......................................... 4
   1.7 Outline ............................................... 4

2 Background Study 5
   2.1 Bluetooth technology .................................... 5
       2.1.1 Bluetooth Low Energy .............................. 5
       2.1.2 Topology ........................................... 5
   2.2 Network Model ........................................... 6
   2.3 Topology Formation ..................................... 7
   2.4 Energy Aware Routing ................................... 9
   2.5 Network Lifetime ....................................... 10
   2.6 Reinforcement Learning ................................ 11
       2.6.1 Markov Decision Processes ....................... 12
       2.6.2 Q-Learning ........................................ 13
   2.7 Routing based on Reinforcement Learning ............... 14
       2.7.1 Q-routing Model .................................... 14
       2.7.2 Q-routing Model handling dynamic networks ... 15
       2.7.3 Q-Value Approximation ............................ 16
       2.7.4 Discount Factor .................................... 16
       2.7.5 Knowledge Exchange ............................... 16
   2.8 Research Gaps ........................................... 17

3 System Design 18
   3.1 BLE Device Design ...................................... 18
   3.2 Transmission Manager and Model Designs ................. 19
4 Methodology
4.1 Qualitative ................................................. 21
4.2 Philosophical Assumption ................................. 21
4.3 Research Methods ........................................ 22
4.4 Research Approach ...................................... 22
4.5 Research Strategy/Design ................................. 22
4.6 Data Collection ........................................... 23
4.7 Data Analysis ............................................. 23
4.8 Quality Assurance ....................................... 23

5 Q-learning Routing Algorithm ............................ 25
5.1 Proposed Discount Factor ................................. 25
5.2 Proposed Reward Function ............................... 25
5.3 Proposed Algorithm ..................................... 26
5.4 Proposed Q-value Approximation ....................... 27
5.5 Period and Threshold ................................... 27

6 Performance Evaluation and Results ..................... 30
6.1 System Parameters ....................................... 30
6.2 Impact of Learning Rate and Discount Factor Weight 32
6.3 Impact of Control Packet Period and Threshold Level 33
6.4 Simulation Result ......................................... 34
6.4.1 35 Devices ............................................ 34
6.4.2 55 Devices ............................................ 34
6.4.3 75 Devices ............................................ 36
6.4.4 Use Case ............................................ 36

7 Conclusions and Future Work ............................ 37
7.1 Discussion and Result Evaluation ....................... 37
7.2 Future Work ............................................. 38

Appendix A Impact of Learning Rate and Discount Factor Weight 41
Appendix B Impact of Period and Threshold 45
Appendix C Impact of Routing Algorithms 49
List of Figures

2.1 A piconet consisting of one master node (red) and six slave nodes (blue). ................................................. 6
2.2 A scatternet consisting of piconets interconnected with bridge nodes (green). ............................................ 7
2.3 Markov decision process, agent interact with environment. .......... 12

3.1 BLE Device Design, main module overview. .................. 18
3.2 BLE Device Design, sub-modules overview. .................. 19
3.3 BLE Devices communicate via Transmission Manager. ........ 20

5.1 Example of how often a control pack is sent back to the sending node, in a network with the configuration of Threshold at 10 and Update Period at 20 ......................................................... 28
5.2 Process flowchart for sending a packet. ......................... 29

6.1 Overview of the average results for Learning Rate, Discount Factor Weight, Period and Threshold for 35, 55 and 75 devices. ... 35

A.1 35 Devices, Learning Rate and Discount Factor Weight; 30%, 50%, 70% and average. ........................................ 42
A.2 55 Devices, Learning Rate and Discount Factor Weight; 30%, 50%, 70% and average. ........................................ 43
A.3 75 Devices, Learning Rate and Discount Factor Weight; 30%, 50%, 70% and average. ........................................ 44

B.1 35 Devices, Period and Threshold; 30%, 50%, 70% and average. . 46
B.2 55 Devices, Period and Threshold; 30%, 50%, 70% and average. . 47
B.3 75 Devices, Period and Threshold; 30%, 50%, 70% and average. . 48

C.1 Comparison of different Routing Algorithms on 35, 55 and 75 devices. The Average Node Degree and coin-cell percentage is displayed on each topology. .............................. 50
C.2 Use case, 100 Devices, 34% coin-cell .......................... 51
List of Tables

2.1 Characteristics that could be considered for RSM calculation. . . . 8
2.2 Summary of edge colors used during topology formation. . . . . . 9
2.3 Markov Decision Process. . . . . . . . . . . . . . . . . . . . . . . . 13

6.1 Summary of parameters in the simulation environment. . . . . . 31
6.2 Summary of parameters in use case simulation environment. . . . 31
6.3 Summary of the best performing parameter configuration for ev-
er every number of device with each coin-cell density, found in Section
6.2. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 33
6.4 Summary of the best performing parameter configuration for ev-
er every number of device with each coin-cell density, found in Section
6.3. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 34
Acronyms

**BF** bandwidth factor. 16

**BLE** Bluetooth Low Energy. 1–7, 10, 11, 17–19, 38

**BSF** Bluetooth Scatternet Formation. 7

**BSF-UED** Bluetooth Scatternet Formation based on Unnecessary-Edges Deletion. 7, 9

**CiD** characteristic identity. 8, 9

**DP** Dynamic Programming. 13

**EAR** Energy Aware Routing. 3, 4, 9, 19, 30, 32, 34, 36, 37

**GPI** Generalized Policy Iteration. 13

**IoT** Internet of Things. 1, 3, 5

**M2M** machine-to-machine. 1

**MANETs** Mobile Ad Hoc Networks. 16

**MARL** Multi-Agent Reinforcement Learning. 14

**MDP** Markov Decision Process. 12

**MF** mobility factor. 16

**ML** Machine Learning. 11

**PF** power factor. 16

**POMDP** Partially Observable Markov Decision Process. 14

**QoS** Quality of Service. 16
RL Reinforcement Learning. 2–5, 11–14, 16, 17, 21, 22, 37, 38

RSM Role Suitability Metric. 7–9, 20

SP Shortest Path. 3, 14–16, 19, 30, 32, 34, 36, 37

TD Temporal-Difference. 13

TFRS Topology Formation considering Role Suitability. 7, 8

TxRx transmission/reception. 18, 19

UiD unique identifier. 8

UWSN Underwater sensor network. 16

VANETs Vehicular Ad Hoc Networks. 17

WSN Wireless Sensor Network. 17
This page intentionally left blank.
Chapter 1

Introduction

As the society is moving towards a more technology dependent one with smart cities, smart homes and smart offices, the demand on connecting a large number of devices increases. The IoT paradigm arises from this increasing number of connected devices, involving the M2M paradigm for short-range communication in the networks [1]. In [2], Ericsson gives an overview of the functionality that is needed to connect capillary networks, configure, manage and provide end-to-end security. Ericsson states that capillary networks are a promising solution, and to enable it different kinds of short-range communications technologies like Wi-Fi and Bluetooth are proposed. To create and maintain an efficient and reliable IoT network, there are a lot of difficulties and challenges, for example; in a scenario where M2M communication is between battery-operated devices the energy consumption is crucial. Further, the network lifetime is a valid quality metric for the network. This study will focus on intelligent and energy efficient routing in capillary networks to increase the network lifetime. This chapter present background information on the subject, the problem formulation that this study is trying to solve, the purpose of the thesis, the goal with the study, used methodology and delimitations of the study.

1.1 Background

Capillary networks are local networks that uses short-range radio-access technologies to locally connect devices. In previous works, the short-range communication technology used in the capillary network is BLE [3], [4]. Capillary networks take advantage of the key capabilities of cellular networks, described in [2]. BLE provides low cost and energy efficient operations compared to ZigBee. The result in the study, [5], show that BLE, in terms of number of bytes transferred per Joule spent, is more energy efficient. However, the technology comes with limitations, it allows only direct communication between master and slave devices.

One major problem with IoT networks is the heterogeneity of the devices
in the network. The devices could differ in terms of energy, e.g. connected to mains or have a limited energy resource (battery). Device heterogeneity could also include differences in memory, computational capabilities or mobility (stationary or mobile). Device heterogeneity becomes a challenge because it needs to be taken into consideration when routing packets in the network. Traditional routing approaches assume homogeneous devices, and therefore, a simpler approach works when selecting for example the minimum hop route. In this case, the heterogeneity of the devices could lead to lower performance of the route that a traditional routing approach chooses if it consists of battery powered devices or devices with low computational capabilities. More, this could result in lower network lifetime if the battery powered devices die or higher transmission time due to low computational capability on the devices.

Considering a real-life scenario in a smart home or a smart office, different devices would be homogeneously deployed. For instance, different kinds of sensors or printers are stationary and often homogeneously distributed. Take temperature sensors as an example, they are placed in the office or home to cover as much area as possible to give an accurate measurement hence symmetrically disposed.

1.2 Problem

The quality of a mesh network constructed by multiple heterogeneous devices can be defined as lifetime of the network. A lifetime has a broad definition and even if it may differ between networks, the most general definition according to [6] is the total time in which a network is operational. Operational, means in this context, when the network is still able to send packets from source nodes to destination nodes. Thus, an important part of a network is to have a long lifetime. This thesis will focus on the possibility to increase the lifetime of a network using a more intelligent routing algorithm, applied on a BLE mesh topology, developed with RL. An advantage of using a routing algorithm based on RL is the ability to learn the network by interacting with it. No prior knowledge of the topology is needed and therefore some energy can be saved due to no extra cost in updating every nodes information about the topology. The RL approach is important because it could adapt to changes in the network, such as the battery level in our case. An RL routing algorithm is constantly learning and updating its values, hence it could increase the lifetime by making intelligent routing choices. A main drawback of the approach is that the routing algorithm will have a learning phase. Thus, before it has learned how to act at certain states, it first needs to learn the network by interacting with it. This takes time in the beginning of the network lifetime and will reduce the performance of the algorithm. Our aim in question form:

"How can a distributed Reinforcement Learning routing algorithm positively affect the lifetime of a Bluetooth Low Energy mesh network with heterogeneous devices?"
1.3 Purpose

This report will describe a solution for RL routing in BLE mesh IoT networks. Then, it will examine the performance of the solution and compare it against two reference algorithms, Shortest Path (SP) [3] and EAR [7]. Finally, it will discuss the results and define a conclusion.

1.4 Goal

Our goal is to develop and test the use of a RL routing algorithm to examine if it can increase the network lifetime. The focus will be to develop a new distributed routing algorithm using RL, with the main objective to make the network stay operational longer and thus achieve a better lifetime. The results will be used to evaluate the solution developed in this project. It will consist of data from several simulations that will be presented in tables and graphs and compared to the two reference algorithms SP and EAR where the simulations are on the same formation and properties [3], [4].

1.4.1 Benefits, Ethics and Sustainability

Those who will directly benefit from this project are first and foremost people who research and work in the area. The material provided by this project could be used for further development of an optimal routing algorithm but also a more sustainable network. Eventually it could benefit anybody that uses or works with a network of this type. Smart homes and offices is something that are getting progressively more established in today’s society. Products released today are more technologically equipped than ever before. Thus, the ability for products to connect to each other today, has never been this universal before. A trend that does not seem to be terminated in the future either, but rather will increase. As long as there is a demand from society for smart-products, the market will deliver. With that, the number of products that goes under the category IoT will grow, and the importance of a good sustainable environment for connecting and sending data between them is crucial. This project would then contribute positively for a more sustainable network and a longer network lifetime would benefit the users.

1.5 Methodology

Our study has five major phases, the first, the literature study and background research helped us to understand related works and concepts in the area of mesh IoT networks and RL. It also helped to get a good understanding of the simulation environment and system that was used in related works [3] and [4]. Second phase, design a solution that could be applied in the system and could be compared against the reference algorithms in the system. The third phase consists of the actual implementation of the algorithm. Phase four, test and simulate the
algorithm in the simulation environment, e.g. produce results. The final phase includes analysis of the results and a comparison against reference algorithm. All the phases after the literature study are described in greater detail in Chapter 4, stating our research methods and methodologies. Summary of the study phases:

1. Literature study and background research.
2. Design a solution.
3. Implement the algorithm.
4. Run the algorithm in a simulated environment.
5. Analysis and comparison.

1.6 Delimitations

The simulation environment is one of the delimitations in this study, because it has a simplified model to make the results deterministic and comparable. If more realistic models were used it would increase the complexity of the work, but also make it more like a real-life scenario. The mobility of the devices is also a delimitation, in a real-life scenario the devices would probably move around as time proceeds, however in the simulation they all have a static position. The device heterogeneity is also limited by certain boundaries all explained in Chapter 3. The study is also limited to use only one approach to the RL routing algorithm solution. A more detailed study could be done using different approaches, to find the one best suited for routing in BLE mesh networks.

1.7 Outline

Chapter 2 presents the background study. It includes a short review of Bluetooth technology and related work on topology formation. Followed by a short introduction to one of the reference algorithms, the EAR algorithm. An explanation on the network lifetime is given, and at the end a brief introduction to RL and related work in routing based on RL are presented. Chapter 3 describe our system design and it involves describing the simulator parameters and settings. Chapter 4 present the methodologies and methods used in the study. Chapter 5 presents the proposed algorithm and the work that was done to find a solution. Chapter 6 presents the performance of the proposed algorithm compared to reference routing algorithms. Finally, in Chapter 7, conclusions are described, positive effects and drawbacks are discussed and at the end possible future work are presented.
Chapter 2

Background Study

First, this chapter will start with a short review of the Bluetooth technology and also a description of the network model. Second, related work on the topology formation algorithm used in our works will be presented followed by a short description of an energy aware routing algorithm applied on the same environment. Third, the concept of network lifetime is presented. Fourth, a brief introduction to RL is given, followed by a summary of related work on routing based on RL. Lastly, research gaps are stated.

2.1 Bluetooth technology

A short technical review and description of the Bluetooth technology is described in this section. The topology of a Bluetooth mesh network is also briefly introduced.

2.1.1 Bluetooth Low Energy

The BLE technology is used for short-range wireless communication and it uses the 2.4 GHz radio frequencies. The BLE technology intend to reduce the power consumption and cost but maintaining a similar communication range as Classic Bluetooth. Recent improvements have been done to the BLE technology through the release of Bluetooth 5. The range have been increased by a factor of 4, which leads to a more robust and reliable connection. IoT networks in smart homes and offices will benefit a lot because of the increased range and makes the Bluetooth technology an even more valid choice for IoT networks. Bluetooth 5 also doubles the speed of data sent and increases broadcasting capacity [8].

2.1.2 Topology

A BLE mesh network has a star like topology with smaller piconets interconnected to form scatternets. Piconets consist of one device with the role of master
Figure 2.1: A piconet consisting of one master node (red) and six slave nodes (blue).

and slave nodes connected to the master, see Figure 2.1. Master node is marked as a red node and slaves as blue nodes. The slaves have no direct communications with each other, therefore the master nodes need to forward traffic between nodes. A slave node can be a part of multiple piconets, therefore have multiple masters and acting as a relay between piconets. The interconnection in BLE mesh networks between piconets can be established through three types of interconnections, one-hop, two-hop and three-hop interconnects, forming the scatternets. The nodes that connect the piconets are called bridge/relay nodes. Figure 2.2 shows a connected scatternet, the bridge nodes are marked with a green color, master nodes with red and slaves with blue.

2.2 Network Model

The network model is similar to a real-life situation as an office or a smart home, where multiple devices such as, climate sensors, coffee-machines, printers, smart locks or smart phones, needs to be connected. According to these different types of devices, they are divided into three types of power sources, mains, rechargeable battery and coin-cell battery. The devices are homogeneously distributed in the network area, i.e. the mains connected, rechargeable battery and coin-cell battery devices are evenly disposed according to their characteristics. To represent the office or home environment the positions of the devices are distributed on a Euclidean plane with an area of $30 \times 30$ meters. Every device has a set maximum communication range of 10 meters, where packets are forwarded on a loss-less and collision-free channel between devices. We use the same network model as the related work [3] and [4].
2.3 Topology Formation

The topology formation is the process where the devices in the network connect to each other to create the network topology. This part is important in a BLE mesh network because of the different roles that needs to be assign and their high impact of network lifetime. In related works, an algorithm called Topology Formation considering Role Suitability (TFRS) is proposed for creating the topology in a BLE mesh network with heterogeneous devices [3]. The TFRS algorithm is based on the algorithm Bluetooth Scatternet Formation based on Unnecessary-Edges Deletion (BSF-UED) presented in [9], which is an enhancement of the BlueStars algorithm [10]. BSF-UED is derived from the Bluetooth Scatternet Formation (BSF) problem which in turn is derived from the specifications and design of the Bluetooth technology. BSF-UED main contribution over other BSF algorithm is the short execution time. The algorithm is deterministic in forming connected scatternets, but to make the scatternets outdegree limited, i.e. maximum 7 slaves per piconet, heuristics are added to BSF-UED. Their experiments showed a close to optimum solution when looking at the outdegree limit of the formatted topologies.

The work, [3], introduces a new metric called Role Suitability Metric (RSM), that is used for forming the topology in the network. RSM summarizes device characteristics with associated weights to create a scalar value for each device which can be compared against other devices when selecting roles in the topology. See Equation 2.1.
Device Characteristics | Explanation
---|---
Energy | Current energy level
Mobility | Probability to stay in a given piconet
Traffic Class | Half-duplex UL, half-duplex DL or full-duplex
Power Class | Supported power class
Memory | Memory available
Compute Capability | Processing power

Table 2.1: Characteristics that could be considered for RSM calculation.

\[
RSM = \sum_{i=1}^{n} c_i \cdot w_i \tag{2.1}
\]

where \( c \) is the device characteristic, \( w \) is the associated weight, \( n \) the number of device characteristics used and \( i \) is the current device characteristic. For example, if the RSM value uses one device characteristic, \( c_1 \) correspond to the value of that characteristic and \( w_1 \) the associated weight to that device characteristic. In the related works, [3], they propose two different approaches to compute the RSM, see Equation 2.2 and Equation 2.3. Equation 2.2 only consider the energy levels and Equation 2.3 added information about the number of neighbors \( N \) and an associated weight \( W \), \( W \) is set to 3.

\[
RSM_{E} = \begin{cases} 
200, & \text{Source = Mains Supply} \\
EneLvl, & \text{Source = Rechargeable} \\
0.1 \cdot EneLvl, & \text{Source = Coin – cell}
\end{cases} \tag{2.2}
\]

\[
RSM_{EN} = \begin{cases} 
200 + N \cdot W, & \text{Source = Mains Supply} \\
EneLvl, & \text{Source = Rechargeable} \\
0.1 \cdot EneLvl, & \text{Source = Coin – cell}
\end{cases} \tag{2.3}
\]

\( EneLvl \) defined in Equation 2.2 and 2.3, correspond to the energy level of a battery powered device and can vary between 1-100%. Hence, the mains supply connected devices will have a higher RSM value \( (RSM \geq 200) \), than the battery powered devices. A summary of device characteristics mentioned in the related work is shown in Table 2.1 together with a short description of the characteristics.

The TFRS algorithm are divided into two phases. Phase one consists of piconet construction and the second phase of piconet interconnection. In the first phase piconets are constructed using the RSM metric to select the best apted nodes for master roles. To break ties each node has a comparable unique identifier (UiD); in case of the exact same RSM value the node with highest unique identifier is selected. The UiD together with the RSM value form the tuple, characteristic identity (CiD), shown in Equation 2.4.
The current state of a node $u$ is denoted by $\text{state}(u) = \{\text{none, master, slave}\}$. The state of an edge between a node $u$ and a node $v$ is denoted by $c(u,v) = \{\text{black, silver, green, red, blue}\}$. The colors is defined in [9] and summarized in Table 2.2.

The process starts in a distributed manner and each local maximum considering the CiD start capturing nodes. Edge colors keep track of the process and it continues until every node belong to a piconet.

In the second phase, the piconets are interconnected using defined interconnect rules and prioritizing rules for selection gateways. The piconets are treaded as individual meta-nodes during the interconnection and the master of each piconet have a gateway table. The gateway holds information of neighbor piconets. In the works, [3], two methods are used to select a gateway. The first one is based upon the base algorithm BSF-UED, but they included breaking ties using energy level. The second method is a completely new approach were only the energy levels are used when selecting the gateways.

The gateways have a different terminology mentioned in the works [4] where they are called bridges. More work has been done with the RSM metric presented earlier. To enable a more resilient network a secondary bridge between the piconets is introduced [4]. The simulations show an increasing lifetime when adding a secondary bridge combined with RSM based master and gateway selection.

### 2.4 Energy Aware Routing

Extending the works of [3] and [4], an EAR algorithm is proposed to increase the lifetime in the RSM based topology formation [7]. The EAR is an centralized and static routing algorithm, which means that when a route from source to destination have been selected it will never change. Information collected by nodes in the network are available by all other nodes. To discover routes, a
flooding-based route discovery is used. To select a route, the algorithm calculates weights for all nodes in every discovered route from source to destination, the weight of the node is the battery level of the device. It then selects the one with \( K \) minimum weights. The weights \( w \) are calculated as follows in Equation 2.5.

\[
w_{i,j} = E_{i,j}^{max} - e_{i,j}
\]

\( E_{i,j}^{max} \) denotes the maximum energy level of a node \( j \) in route \( i \) and \( e_{i,j} \) denotes the residual energy of a node \( j \) in route \( i \). This approach results in more load balancing in the network, because of the selection of higher residual energy level in the devices in selected routes.

### 2.5 Network Lifetime

Networks constructed by some or only battery-powered devices could also consider the networks lifetime as a performance metric. It is a more valid performance metric for networks consisting of battery-powered devices unlike networks consisting of devices with unlimited power. Measuring a networks lifetime can be done in multiple different ways, and neither of them is right or wrong. It all depends on how the topology of the network looks like and the circumstances of network usage. If it carries important information or every device is of significant importance, then a network lifetime may possibly be measured by the time it takes until the first device dies in the network. In contrast, a network that can still work after a device dies, could have its lifetime measured as when a fraction of all the devices have died or the success rate of delivered packages falls under the acceptable threshold. Everything depends on the network design, its purpose and the importance of each individual node. A networks lifetime can thus be defined as the time that a network can withhold the expected requirement.

In addition to the previous mentioned measuring methods, there is another one, where the devices can be categorized into two groups; critical and non-critical devices [11]. How this categorization is done differs between network to network. A device categorized as a critical, would intend that it has a more important status than other devices in the network. The more important status could indicate that if it died, the network would stop working. In contrast to a device with a less important status would be non-critical, where if it dies, the network would not be affected and could still work. In some networks all devices could have the same status, and there would not be any critical and non-critical categories.

For this study, the topology that will be used is heterogeneous Bluetooth devices connected in a mesh network. Considering the different roles in the BLE topology, it is suitable to categorize slave nodes as non-critical and master and bridge nodes as critical. A good method to use for measuring the network lifetime with this kind of set can then be to measure when the first critical device dies or when a percentage of the critical devices dies. The method that will be used in this study is measuring when the first critical device dies.
A device dies when it runs out of energy, so to increase the network lifetime it is important to understand what affects the energy usage of devices and how to make every single device live as long as possible. A packet sent through a device has a price in form of energy taken from the device it passes through. Therefore, a device that has traffic heavy is more likely to die first than the ones that are used less frequently. For example, a master node is expected to have a much higher traffic flow of packets than a slave node. Because of the BLE topology that is being used in this work, specifically two different decisions affect the network lifetime the most, role selection and the routing algorithm. In previous works, the selection of the roles were investigated and different algorithms for choosing the optimal roles for every device was developed [3], [4]. There are three types of the heterogeneous devices, one that is connected to power, a rechargeable battery powered and the battery powered device. Determining the role of each device in the network will have a distinguishable effect on the networks lifetime. To assign the master role to a battery powered device can be inefficient if another device which is connected to the power grid is available. Also, evaluating and changing; if necessary, these roles during the network’s life is something to consider in order to increase the lifetime. Research on the impact of the routing algorithm used in the network has also been done in previous work [7]. They concluded that developing an algorithm for routing that makes decisions with respect to the residual energy of devices could increase the network lifetime. Instead of only taking the shortest path to the destination node, the algorithm considered the energy left in the nodes. More details of these works are described in Section 2.4.

2.6 Reinforcement Learning

RL is performed by an agent from interaction with its environment. The agent does not rely on supervision or complete models of the environment; hence it is learning by interacting. RL is a computational approach for understanding and making decisions to achieve long-term goals. It can be anything, such as finding the fastest way out in a maze or by getting the highest score in the game Pacman. The long-term goal is very individual but can be summarized as getting the maximum reward out of a task. Compared to Machine Learning (ML), more precisely supervised learning and unsupervised learning, RL have a little different approach. Supervised learning is learning from a training set of labeled (specification about the correct action, specified by a external supervisor) examples, with the goal of generalize its responses and be able to react on situations outside the training set. In this case, RL are different in the way its learns by interacting directly with the environment (no training beforehand). Unsupervised learning is about finding structure hidden in unlabeled data collections. RL differs from unsupervised learning in the way it is trying to maximize the reward not finding hidden structure. Considering the differences between the paradigms, Ricard S Sutton and Andrew G Barto consider RL as a third ML paradigm [12].
Three main elements of RL is policy, reward signal and value function. The policy defines the behavior of the agent at a given time, what action to take in a given state. A reward signal is the immediate signal the agent gets from the environment, defining the goal in a RL problem. The value function defines what is good in the long run, taking states that are likely to follow into account. This helps the agent not only prioritizing immediate rewards but also try to achieve a long-term reward.

2.6.1 Markov Decision Processes

![Markov Decision Process Diagram](image)

Figure 2.3: Markov decision process, agent interact with environment.

The formal framework of a Markov Decision Process (MDP) is used to formalize the problem of RL. In a MDP, at every time step $t$, the environment is in a state $s$ and when the agent takes an action $a$ it receives a reward $r(s, a)$. See Figure 2.3. The agent seeks to maximize this reward over time. Once the agent has taken an action and the environment gives the agent a reward the environment changes to a new state $s'$. The change in the environment can be described with a transition model, that can be formulated as a stochastic transition probability. The agent can therefore also be described as a learner and a decision maker that aims to maximize the reward value over time [12].

A summary of the components of a MDP, and notations are shown in Table 2.3. The state $S$, is a representation of the environment. The model is the probability of changing state, to add even more complexity to the problem a probability of changing state makes it harder for the agent to learn, because it does not know what an action might lead to. Sometimes it might change for the better and sometimes no changes, which makes it harder to know if the action was good or not. An action $A$ is taken by the agent and it will affect the environment, according to what happened the reward $R$ is sent back as feedback to the agent. The policy $\pi$ defines the optimal decisions in a current state. $\pi$ can be seen as the solution to the MDP problem.
Components | Notations
---|---
State | $S$
Model | $T(s, a, s') \sim \Pr(s' | s, a)$
Action | $A, A(s)$
Reward | $R, R(s), R(s, a), R(s, a, s')$
Policy | $\pi$

Table 2.3: Markov Decision Process.

### 2.6.2 Q-Learning

RL has multiple different approaches, some of which are Dynamic Programming (DP), Monte Carlo methods and Temporal-Difference (TD) learning [12]. All of them use some form of Generalized Policy Iteration (GPI). TD learning combines ideas from Monte Carlo methods, learn from raw experience, and DP methods, update estimates based on already learned estimates. Our work will focus on TD learning, more precisely an approach to TD learning called Q-learning [12]. The Q-learning function is defined by Equation 2.6.

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha [R_{t+1} + \gamma \max_a Q(s_{t+1}, a) - Q(s, a)] \quad (2.6)$$

where $\alpha \in (0, 1]$ and $\gamma \in (0, 1]$. $Q^*(s, a)$ defines the optimal Q-learning function, which is directly approximated by Equation 2.6. To maximize $Q^*(s, a)$, and in that way follow the optimal policy $\pi^*$, the optimal action $a^*$ at a given state $s_t$ are required to be identified. $a^*$ is defined by Equation 2.7.

$$a^* = \max_{a_t \in A_{s_t}} Q(s_t, a_t) \quad (2.7)$$

The following pseudo-code shows a generalized Q-learning algorithm for estimating the optimal policy. Each episode has a fixed set of time steps $t$. See Algorithm 1 for an overview of the process.

```plaintext
Arbitrarily initialize $Q(s, a)$ for all $s, a$;
for each episode do
    Initialize state $s_t$, $t \leftarrow 0$;
    for each step $t$ of the episode do
        Choose action at a given state $s_t$ according to Equation 2.7;
        Take action $a_t$, observe reward $r_t$ and state $s_{t+1}$;
        Update $Q(s_t, a_t)$ according to Equation 2.6;
        Update state $s_t$ as follows;
    end
    $s_t \leftarrow s_{t+1}$
end
```

**Algorithm 1:** Generalized Q-learning algorithm.
2.7 Routing based on Reinforcement Learning

The first implementation of RL routing in networks was first introduced in 1994 by Michael Littman and Justin Boyan. They formed the first model for routing schemes in distributed wireless networks, the Q-routing model. The work they accomplished would establish the foundation for using RL in routing. Since then multiple different versions of RL routing models have been presented. The most acknowledged are Multi-Agent Reinforcement Learning (MARL) model and Partially Observable Markov Decision Process (POMDP) model, but they are not considered in our study and we will only focus on the Q-routing model.

2.7.1 Q-routing Model

The two researchers Michael Littman and Justin Boyan from Carnegie Mellon University School of Computer Science wrote a paper on a self-adjusting algorithm for packet routing in a network [13]. Where the main goal was to sustain the highest possible level of traffic, on an irregular topology network with unpredictable usage patterns. Thus, the performance of the routing policy is measured by the total time it takes for a packet to be delivered to the destination from the source.

They used a variation of a Q-learning algorithm that consisted of multiple distributed learners, one in each node. Every learner has a Q-table where it stores an estimated value of how long it takes to send a packet from that current node \( x \) to destination node \( d \), via an specific neighbor node \( y \). The learning scheme then selects the node \( \bar{y} \) that acquires the lowest estimated delivery time of all neighboring nodes. After the packet is sent, it instantly updates its delivery estimated value by getting information from the receiving node. Since that node is presumably closer to the destination node, it is considered to give a more accurate estimate on how long it will take to deliver the specific packet. The estimated time for node \( \bar{y} \) to deliver the packet to \( d \) will be \( Q_{\bar{y}}(\bar{z},d) \), where \( \bar{z} \) is the node from \( \bar{y} \) with the best estimated time for delivery. The value is then updated by multiplying a learning rate \( \eta \) with the change of the new estimated delivery time and the previously estimated value. The new estimated delivery time is the sum of the value from the selected node plus the time the packet would spend in the queue \( q \) (if there is a queue) at the selected node. The nodes can only handle a certain number of packages each time unit, which means, if more packets arrive at a node than it can handle, they are placed in a queue. Littman’s and Boyan’s Q-function is defined as in Equation 2.8.

\[
\Delta Q_{x}(\bar{y}, d) = \eta(Q_{\bar{z}}(\bar{z}, d) + q - Q_{x}(\bar{y}, d)) \tag{2.8}
\]

Littman and Boyan concluded that after testing the algorithm on a variety of grid network topologies and comparing it to a SP algorithm it could sustain a higher level of network traffic than what the SP could. The Q-learning algorithm managed bottlenecks in the network more efficiently than the SP one. Thus, it could handle a situation when the network where under a high load of packets.
Q-routing Model handling dynamic networks

Some years later Littman and Boyan released a new paper [14], where they had continued the work. This time three different specific situations with the focus on dynamically changing networks were examined and tested. First, they considered a network where the topology changed and links between nodes were manually disconnected during simulation. The Q-routing algorithm reacted quickly to the changes and manage to route packets efficiently after the disconnected links.

The second situation that was tested was a network simulation that changed back and forth periodically between two different routing patterns. One pattern was between upper and lower part of the irregular grid and the other was between left and right side of the grid. Again, the Q-routing algorithm reacted as intended and could after a short time of inefficient routing adapt to the new situation and transmit the packets on the most optimal route.

The last case tested was a simulation where the networks load level of the packets changed periodically. As before demonstrated the algorithm can handle a network when exposed to packets sent at a low rate and will then choose the SP. Increasing the load to a higher level is also shown in previous study that the algorithm can control and adapt to the new bottlenecks. When testing on dynamically changing load levels the algorithm will not adapt quickly when changing back from a high level to a lower level of packets stressed on the network. It will not choose the most optimal SP again for routing packets after the load have been decreased from a high level.

This is because of the Q-learning algorithm is a greedy algorithm and only updates the values of the best neighboring nodes and are therefore not able to discover shortcuts. An optimal route may not be chosen even if it would have the lowest delivery time of the available routes. A node may learn to overestimate the delivery time of an optimal route, which can happen when the load of the packets is changing in the network. Then as long as a suboptimal route have a lower estimated value it will be chosen instead. Several so-called exploration techniques have been proposed to fix this problem and find the “hidden” shortcuts. Such as implementing a static or probability to take a suboptimal route or by using a function to get updated values from its neighbors at a rate [12]. One of the most established methods is the use of randomness in the learning state. Let the algorithm have the possibility to take the suboptimal route some random amount of times. This approach leads to two drawbacks, the initial period of exploration will never end because of the network never stops changing and random traffic have a negative effect on the congestion of the network. A method that can be used to avoid these drawbacks is to implement a modification to the Q-routing called full echo. It sends out a request signal; on a separate channel, to all neighboring nodes every time it is about to send a packet. The nodes response with a number of their estimated delivery time of the packet to destination. Those numbers are then used to adjust the Q-values of the current node to every neighbor.

Littman and Boyman implemented the full echo modification to their algo-
rithm and the result they got shows that it will be equally as good as the SP policy when exposed to a low load and will perform better under a higher load. But it will not outperform the basic Q-routing algorithm.

2.7.3 Q-Value Approximation

There are multiple versions of the Q-learning approximation function that were mentioned in Section 2.6.2. One approach that was studied in [15] uses the function in Equation 2.9 for approximation the Q-values. The proposed routing protocol aims to increase the network lifetime of Underwater sensor network (UWSN) and takes the residual energy of devices into consideration.

\[
Q(s_t, a_t) \leftarrow (1 - \alpha)Q(s_t, a_t) + \alpha[r_t + \gamma \max_a Q(s_{t+1}, a)]
\] (2.9)

where \(\alpha \in (0, 1]\), \(\gamma \in (0, 1]\) and \(r_t\) is the reward function. The reward function is based on two residual energy parameters. The first one uses the residual energy of the node and the other uses the average residual energy of a group of neighbor nodes.

2.7.4 Discount Factor

In [16], a routing protocol for Mobile Ad Hoc Networks (MANETs) is proposed. The protocol is based on Q-learning considering improvement of Quality of Service (QoS) parameters, such as bandwidth efficiency, link stability and power metric. The main features of the approach are to establish routes which are less likely to fail, i.e. high robustness. The discount factor, \(\gamma\), is used. The estimation of \(\gamma\) is based on three factors from the next-hop neighbor: mobility factor (MF), bandwidth factor (BF) and power factor (PF). MF estimates the lifetime of a link between two nodes, BF represents the available bandwidth and PF is the residual energy of a node. See Equation 2.10 for the calculation of \(\gamma\).

\[
\gamma = \omega \cdot \sqrt{MF \cdot BF \cdot PF}
\] (2.10)

where \(\omega\) is a predefined value, \(0 < \omega < 1\). The main contribution of this approach is that the discount factor becomes dynamic because of the changes in the parameters MF, BF and PF when updating. To compare with the traditional RL approach that has a static discount factor, the dynamic discount factor aims to provide a more accurate estimation on the Q-values of next-hop neighbor nodes. The Q-value approximation used in, [16], is similar to the one in Equation 2.9 and \(\gamma\) is used in the same way, hence the preference on the future long-term reward indicated by \(\gamma\), will change.

2.7.5 Knowledge Exchange

A key enabler for making RL routing a possible routing approach is the exchange of knowledge between nodes. To learn and interact with the network the agent needs to receive information from other nodes in the network. A distributed
RL approach using hello messages for exchanging information in Vehicular Ad Hoc Networks (VANETs) is proposed in [17]. The hello messages are periodic to decrease the control overhead, and the period was set to 1 second. In this work, the nodes attach their max Q-values and two other factors for calculating new Q-values to their hello messages.

2.8 Research Gaps

The use of RL for routing in networks is nothing new, multiple algorithms have over the years been developed with the goal to get the best possible performance for different situations. The previous research has one thing in common, little to nothing have been done in the field of BLE mesh networks with heterogeneous devices. Most of the algorithms have been applied to networks types like Wireless Sensor Network (WSN) and Wireless Ad-hoc Networks. The work has focused on a field that have been left out, where the status and the role of individual devices has a much more substantial impact than devices in WSN networks for example. We also continued the work from the previous studies [7] where the routing algorithm in the BLE mesh network where static and centralized and our approach is dynamic and distributed.
Chapter 3

System Design

This chapter introduces the system design used in our research. The system design is built upon multiple modules and sub-modules, each given a high-level description in this chapter. The models are based on the related work done in [3], [4] and are in some cases defined to simplify the simulation and make the model deterministic.

3.1 BLE Device Design

The BLE devices consist of three main modules, each of them with their own sub-modules. Figure 3.1 gives an overview of the main modules and Figure 3.2 show a more detailed overview of the respective main module with their sub-modules included. The BLE devices have three energy classes, connected to mains, rechargeable battery and coin-cell battery. Connected to mains have unlimited access to power. The rechargeable devices have a larger battery size than coin-cell devices, even though it is rechargeable the probability of being recharged during the simulation is 0 (to achieve deterministic results). The battery level is initialized between 50-99%.

The three main modules are the middleware module, characteristic module and the transmission/reception (TxRx) module. As shown in Figure 3.2a the middleware module includes two sub-modules, topology control block and rout-
The topology control block holds the algorithms for creating the topology formations used in the related work [3], [4] and the routing control block holds the algorithms (SP, EAR and Q-learning routing) for finding the paths in the topology.

In Figure 3.2c, the five sub-modules of the characteristic module are displayed. Each of the sub-modules defines the unique features of the BLE device. The most interesting characteristic sub-module with respect to our work is the energy characteristic sub-module, were information about power source (one of the three energy classes mentioned above), residual energy and battery capacity (if not a connected to mains device). Also, the energy is decreased here when the device is sending and receiving packets. If the residual energy goes below 0 the device is considered non-operational, i.e., the device is dead and can no longer be used to transmit/receive packets.

The last module, TxRx module have two sub-modules. Sending packets are handled by the transmit module and receiving packets are handled by the reception module. See Figure 3.2b.

![Diagram of BLE Device Design, sub-modules overview.](image)

### 3.2 Transmission Manager and Model Designs

The transmission manager enables the communication between the BLE devices providing a transparent channel for transmission of the data. Figure 3.3 illustrates BLE Device 1 transmitting a packet and BLE Device 2 receiving a packet, the communication between all devices are handled in this way. A transparent channel means a channel without collisions and interference; hence the model is
deterministic. The channel is assumed to have infinite capacity and every packet transmitted is received without any fail if the devices could reach each other. We assume an energy cost of 0.1 \( mJ \) for each transmission and reception. To maintain fairness of the traffic model, each device generates an equal amount of data irrespective of the routing algorithm and number of devices. The topology formation algorithm is constant and set to the best-known configuration from the related work [3] and [4]. Best known settings use the RSM metric considering energy and neighbors for master selection and gateway selection considering energy and number of bridges are set to 2.

The routing model from the previous work [3], [4] and [7] can be described as follows:

- Every source node floods control packets thought the network to discover all possible routes.
- When a control packet reaches destination they are acknowledged by sending a response message to initiator for each of the different path it finds.
- The algorithm chooses the path according to minimum hop or energy aware criteria.

The routing model for the Q-learning does not have the initial phase of flooding packets to discover route since it is distributed. The agent instead explores all the routes and receive a control packet after each hop with the data needed to update the corresponding Q-value. We consider the cost of this control packet the same as a regular data packet. Even if this control packet is much smaller in size and the transmission cost would in a real-life scenario probably be much less than the transmission cost of a regular packet. However, this is done to make it as fair as possible for the other two algorithms. In the routing model from previous work, the cost for the control packets and the response packets that are flooded have the same cost as a regular data packet.

![Figure 3.3: BLE Devices communicate via Transmission Manager.](image-url)
Chapter 4

Methodology

When selecting the best suitable methods and methodologies for our study we used the portal of methods and methodologies created by Anne Håkansson [18]. The portal consist of seven layers, each of them describing a step in the research, were you should choose one method or methodology on every layer. Two basic research methods are Quantitative Research or Qualitative Research. Our approach is considered quantitative. There is of course a lot more methods out there but Håkansson have summarized the most common. The layers in the portal are, philosophical assumptions, research methods, research approaches, research strategies/design, data collection, data analysis and quality assurance. The following subsections will go through each of these layers and explain our choices and how they impact our research process. However, the first step is the literature study to get to know the field and make it possible to do appropriate choices of methods and methodologies.

4.1 Quatitative

As mentioned earlier, we have a quantitative research method for our project. A project that is of quantitative character often applies to projects that are numerical. Our study is numerical because of our choice to use RL which is a computational approach to learn as mentioned in Section 2.6. The Q-value approximation consist of different variables that are measured and tested to evaluate their impact on our research question. The network lifetime is also measured and give us a numerical result to evaluate.

4.2 Philosophical Assumption

The philosophical assumption is the starting point for the whole research process. It will state the point of view to the project and can be seen as the stand-point. Our study is based on the positivist paradigm which is a philosophical assumptions that is very suitable for testing performance in our field
The positivist paradigm suits our study very well since it is often used in projects that are of experimental and testing character.

4.3 Research Methods

In the paper by Håkansson [18], she has made a distinction between research methods and research strategies/designs. The research methods are described as being on a higher level than research strategies/design, and can be seen as the framework for the research. We have chosen an Experimental research method. Our project studies effects of the lifetime using a RL routing algorithm based on the Q-learning technique. Our approach uses multiple variables that is changed to evaluate the best performance. We try to find relationships between them by keeping some of them constant while changing one of them.

4.4 Research Approach

This study uses a deductive research approach, which is often used for testing the theories in a project or verify or falsify the hypothesis in a project. According to Håkansson [18], it is almost always used when the project is based on quantitative methods with large data sets to test the theory or hypothesis. Since our study uses quantitative methods and the result is based on multiple simulations a deductive research approach is appropriate.

4.5 Research Strategy/Design

Håkansson defines the research strategies and designs as the guidelines or the methodologies for carrying out the research [18]. We use an experimental research strategy as we described in the above Section 4.3, this approach provide cause-and-effect relationships between variables. However, this section describes it on a lower level, including how we are going to organize, design and conduct our research. After the literature study and study of the simulation environment that we use from previous work, we start with designing a solution. The major steps in designing the solutions is to come up with a RL approach which takes the energy into consideration. Next step is then to implement it in the environment. To figure out the best parameters for our proposed solution we use the experimental research approach were we keep some of the variables constant while trying out different values on others. First, we start by finding the best $\alpha$ and $\omega$ (described in Chapter 5) for the different topologies (different number of devices), by keeping one of them constant while changing the other, testing all possible combinations. When we are done, we will try to find the best possible threshold and update period (described in Chapter 5). Since network lifetime is the only performance metric used the results are easy measurable, and we can decide which values that performs best.
4.6 Data Collection

As briefly described above we use the simulation environment from previous work to collect data. The experiments are carried out with the same simulation settings only changing the routing algorithm when we compare against the reference algorithms. The simulator used in the study is written in Java and is the same as the one used in the related work [3], [4] and [7]. The deployment and source/destination pair are generated by Python scripts also used in the related work. Below follows a step-by-step description of the simulator and its parameters:

1. Choose number of devices in the deployment.
2. Choose the device density.
3. Generate random deployment and source/destination pairs.
4. Choose which routing algorithm to apply.
5. Apply the topology formation algorithm (constant).
6. Apply traffic and transmission model (constant).
7. Start simulation.
8. Log network lifetime.

The simulator has a sequential event-based approach to simulate the network. During the whole simulation the network lifetime is logged after each time step into a separate log file. The network lifetime is logged in time units, were one time unit is the time were every device perform one event (e.g. forwarding a packet).

4.7 Data Analysis

When using a quantitative research approach one of the most common data analysis methods are statistics [18]. Hence, we use statistics to model our data we collect from the simulations. We use statistics to compare the different impact of our variables in our solution and we also use statistics to compare the performance of our solution against the reference algorithms.

4.8 Quality Assurance

As we use quantitative research, with a deductive approach, Anne point out the importance of apply and discuss validity, reliability, replication and ethics [18]. The validity of the results produced by the simulator are tested with different test cases. We use some test cases to see number of packets delivered, average residual energy remaining in the network and also print out and track the
routes that our solution selects. The routes are verified by creating a graph of the deployment used, displaying the topology with all devices and their connections. Reliability is not a big issue in our simulations since the simulation is a deterministic program which have no external devices that measure the energy levels or something like that. The replicability of the research is easy to do if the same source code is used. Because of the high number of Java files and code lines the source code is not included in the appendix but available if requested. Regarding the ethics there is no external participants, no private policies or confidential material involved so there is no problems there.
Chapter 5

Q-learning Routing Algorithm

The current chapter introduces the proposed distributed Q-learning routing algorithm. We explain how a dynamic discount factor is used and how we have defined the reward function. Further, we present our Q-value approximation function. Also, the knowledge exchange between nodes using packets are discussed and a period and threshold for that are proposed. At the end, the distributed Q-learning routing algorithm is explained in more detail.

5.1 Proposed Discount Factor

Our proposed discount factor, $\gamma$ uses the idea presented in Section 2.7.4 but instead of using the same factors we introduced a new factor. We use the residual energy of the next-hop node, $e_y$ and combined with a weight constant $\omega$, where $0 < \omega \leq 1$, it defines our discount factor. See Equation 5.1.

$$\gamma = \omega \cdot e_y$$

The weight $\omega$ is set through testing and analyzing results, more detail about it is presented in Chapter 6.

5.2 Proposed Reward Function

The Q-value approximation function also includes a reward parameter, $R$. Depending on the outcome of the action that the algorithm makes, the reward function can define $R$ with three different values, see Equation 5.2. If the action taken by the algorithm, from a node to another node, is a regular forward move, $R$ is set to zero. That means that there is no reward for just forwarding a packet from one node to another without reaching the destination. But if the packet got forwarded to a node that the packet already has been at, in other words,
the packet has done a loop. Then the reward function gives a penalty for that move, and the penalty is negative one. Last value that $R$ can acquire is if the action leads to the destination for the packet, then the action is rewarded with a plus one.

$$R = \begin{cases} 
1, & \text{if reach destination} \\
-1, & \text{if } y \in \text{already visited node} \\
0, & \text{otherwise}
\end{cases} \quad (5.2)$$

5.3 Proposed Algorithm

Every node has its own agent that holds a Q-table, where every element in the table represents an action to forward a packet from that current node to any destination node via a specific node. All the elements in the table are first initialized to zero. When everything is initialized the network is ready to send packets from source nodes $s$ to destination nodes $d$. When a source node gets signaled to send a new packet or a packet arrives at a node $x$ the algorithm will first examine the newly acquired packet and get the destination information of that packet. After that, it evaluates all of the possible action it can take to forward the packet towards the destination. It does so by iterating over the list of the neighbor nodes $y$. It analyzes each value in the Q-table by looking at the values in position of sending the packet via the neighboring node $y$ to destination node $d$, from node $x$. The pseudo-code for the proposed routing algorithm is shown in Algorithm 2.

The Q-table is represented as a matrix and the neighbor node $y$ and destination $d$ correspond to indexes in the matrix, as a Q-value that is represented as $Q_x(d, y)$ (Q-value at node $x$, via node $y$ for destination node $d$). If the destination node is among the available neighboring nodes, it will be selected. However, if that is not the case, it will select the action with the highest Q-value. To make the algorithm exploratory, it prioritizes actions where these values are zero, thus an action that have never been taken before is chosen. In the beginning, every Q-value will be zero, thus the algorithm will take the first node in its neighbor list. The list is sorted by the address number value for each available node.

When a node is selected to where the packet should be forwarded to, the packet is transmitted to that node. After the packet have been transmitted and delivered to mentioned node, a control packet is sent back to the sending node according to the threshold and period criteria presented below. The process of sending a packet to its next-hop neighbor is described in the flowchart in Figure 5.2. The control packet carries information about its residual energy and a Q-value. Where the Q-value is the best value from that node to the destination node via a node $z$, as in Equation 5.3.

$$Q_{y^*}(d, z^*) = \max_{z \in \text{neighbors of } y^*} Q_{y^*}(d, z) \quad (5.3)$$
$d \leftarrow \text{PacketDestination};$
$x \leftarrow \text{CurrentNode};$
$\text{selectedNode} \leftarrow \text{null};$
$\text{NeighborList}_x \leftarrow \text{Every avaible next-hop neighbor of } x;$
if $d \in \text{NeighborList}_x$ then
\quad return $d;$
else
\quad for each node $y$ in $\text{NeighborList}_x$ do
\quad \quad if $Q_x(d, y) = 0$ then
\quad \quad \quad return $y;$
\quad \quad else
\quad \quad \quad if $Q_x(d, y) > Q_x(d, \text{selectedNode})$ then
\quad \quad \quad \quad $\text{selectedNode} = y;$
\quad \quad end
\quad end
end
return $\text{selectedNode};$

Algorithm 2: Q-learning Routing

5.4 Proposed Q-value Approximation

When a node receives a packet with information about a neighbor node it uses the information to update its Q-table according to our Q-value approximation function. We combine (5.1), (5.2) and (5.3) to define our Q-value approximation function, see Equation 5.4. Our Q-value approximation is based on the works presented in Section 2.7.3.

$$Q_{x_{\text{new}}}(d, y) = (1 - \alpha) \cdot Q_x(d, y) + \alpha \cdot (R + \gamma \cdot Q_y(d, z^*))$$ (5.4)

where the learning rate, $\alpha \in [0, 1]$. The best values for $\alpha$ are determined by simulations and discussed in Chapter 6.

5.5 Period and Threshold

Sending a control packet back every time a device has received a packet with information about the nodes characteristics is very expensive especially when the cost of energy for the control packet is the same as a normal packet. Thus, we implemented a period that dictates when the control packets should be sent back to the sending node to be able to update the Q-value. This approach was inspired by the related works presented in Section 2.7.2, the full echo approach [14], and in Section 2.7.5, with the hello messages [17]. Additionally, a threshold function was also implemented to lower the learning time for the algorithm. A threshold means that the first packets, up to a preset level, that is forwarded
to a node, always triggers a control packet to be sent back with the nodes characteristics. Using only a period the learning time for the algorithm would be very long and the network would be exhausted rather than benefit from it. But, by implementing a threshold for the first packets that are being forwarded to a node the risk of fatigue is minimized. To get an visual idea of how this update period and threshold works, an example are illustrated in figure 5.1. The total amount of delivered packets differentiates significantly from each other as the number of nodes in the network changes. Less devices equals less total amount of packets delivered to nodes. Thus, the optimal value for period and threshold also changes. With simulations, these values for every number of devices are determined, more information in Chapter 6.
Figure 5.2: Process flowchart for sending a packet.
Chapter 6

Performance Evaluation and Results

In order to evaluate the performance of our algorithm, a comprehensive and well executed testing phase is crucial. This chapter will present the parameters used in simulations and the results from them. The former algorithms SP routing and EAR will be used as comparison with our new developed Q-learning routing algorithm.

6.1 System Parameters

For every simulation, a deployment is generated with preset parameters. The parameters determine the number of devices, in our case we tested three different quantity levels 35, 55 and 75 devices. The second parameter sets the density of coin-cell devices. This was also done at three different levels 30%, 50% and 70%. These generated devices are then spread out over a square of 30 × 30 meters. The specific parameters were chosen because of the parameters in the related work. With the same configuration of parameters, a comparison is easier and remains the smooth transition between them. With the two parameters; number of devices and coin-cell density, multiple different homogeneous deployment can be generated. A homogeneous deployment means that the nodes are homogeneously distributed over the area with respect to the device type. In addition to the deployment, different traffic models are generated. A traffic model defines the source and destination node for nodes in the network. Multiple different traffic models can be used on the same network topology. Every source node generates a new packet every time unit and sends it towards its destination node. In our study we observed 25 traffic models on each deployment. At each configuration of the number of devices with the proportion of coin-cell, 10 deployments were generated. Thus, for example with a configuration of 35 devices and 50% coin-cell, 250 simulations are executed with every routing algorithm. A summary of the system parameters is presented in Table 6.1. In the paper
[7], a special deployment was tested. This deployment contained 100 devices, were the devices had a preset position in the network. The positions were set to match a real environment in an office to get a real-life scenario. This use case had 34% coin-cell devices. All of the system parameters for this case is presented in Table 6.2. This was tested by simulating 25 different traffic models on the preset deployment; which means on the same topology the source and destination node changes 25 times. These tests were run with all of the three routing algorithms.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>Device types</td>
<td>Mains, Rechargeable, Coin-cell</td>
</tr>
<tr>
<td>Coin-cell battery size</td>
<td>250mAh</td>
</tr>
<tr>
<td>Rechargeable battery size</td>
<td>2500mAh</td>
</tr>
<tr>
<td>Battery level</td>
<td>50-99%</td>
</tr>
<tr>
<td>Deployment area</td>
<td>30 x 30m</td>
</tr>
<tr>
<td>Tx/Rx Range</td>
<td>10m</td>
</tr>
<tr>
<td>Number of Devices</td>
<td>[35, 55, 75]</td>
</tr>
<tr>
<td>Mains, Rechargeable and Coin-cell Density</td>
<td>[35%, 35%, 30%], [25%, 25%, 50%], [15%, 15%, 70%]</td>
</tr>
<tr>
<td>Traffic Rate</td>
<td>1 packet every time unit</td>
</tr>
<tr>
<td>Channel Model</td>
<td>Loss-less and collision-free</td>
</tr>
<tr>
<td>Routing Algorithm</td>
<td>Shortest Path, Energy Aware Routing, Q-learning Routing</td>
</tr>
</tbody>
</table>

Table 6.1: Summary of parameters in the simulation environment.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>Device types</td>
<td>Mains, Coin-cell</td>
</tr>
<tr>
<td>Coin-cell battery size</td>
<td>250mAh</td>
</tr>
<tr>
<td>Coin-cell battery level</td>
<td>99%</td>
</tr>
<tr>
<td>Deployment area</td>
<td>40 x 50m</td>
</tr>
<tr>
<td>Tx/Rx Range</td>
<td>10m</td>
</tr>
<tr>
<td>Number of Devices</td>
<td>100</td>
</tr>
<tr>
<td>Mains, Rechargeable and Coin-cell Density</td>
<td>[66%, 34%]</td>
</tr>
<tr>
<td>Traffic Rate</td>
<td>1 packet every time unit</td>
</tr>
<tr>
<td>Channel Model</td>
<td>Loss-less and collision-free</td>
</tr>
<tr>
<td>Routing Algorithm</td>
<td>Shortest Path, Energy Aware Routing, Q-learning Routing</td>
</tr>
</tbody>
</table>

Table 6.2: Summary of parameters in use case simulation environment.
6.2 Impact of Learning Rate and Discount Factor Weight

Before evaluating and comparing the Q-learning routing algorithm to SP and EAR the best possible configuration on the learning rate and discount factor weight parameters in the Q-function needs to be set. How we concluded for these values was through testing and evaluating different setups of them. The simulations were performed on a network of 35, 55 and 75 devices at every percent level of coin-cell density. The testing was executed with simulation sets of five deployments with five traffic models on each deployment, were performance was measured by network lifetime. The configuration that managed to get the highest value in network lifetime was then chosen as the best. For determine the average best, values from each percent level with the same configuration was added together and an average was concluded. The highest value in network lifetime was selected as the best average configuration for that number of device. At each percent level multiple simulations set were executed with different configurations on the learning rate and discount factor. In total, 144 simulation sets were executed on each number of devices and coin-cell percentage. The graphs in Appendix A, Figure A.1, A.2 and A.3 present results of individual percent level with changing number of devices. The final average value for all the percentages of each number of devices are seen in Figure 6.1 Graphs 6.1a, 6.1c and 6.1e.

For all number of devices, a lower value on the learning rate looks like the most optimal option with a higher value on the discount factor weight $\omega$ parameter. A configuration of the parameters was then concluded of the average best performing in network lifetime. This was done by summarizing the network lifetime at each configuration between the three percentages and selecting the configuration that gives the highest value. Thus, it is not an average over the concluded parameters at every percentages, rather an average over the network lifetime for all the percentages together. A network with 35 devices has the best configuration on average with a learning rate of 0.1 and $\omega$ set to 0.8. For 55 devices it has the same learning rate but a slightly higher value on $\omega$ of 0.9. Finally, for 75 devices the learning rate went up a bit to 0.2 but the $\omega$ stayed at 0.9. All the results for the number of devices and individual percentage levels of coin-cell devices, is summarized in Table 6.3. When comparing the lowest values to the highest values of the average network lifetime with different configurations on learning rate and $\omega$, there is up to 60% change. Thus, selecting the best configuration is very important for optimizing the network lifetime.
### 6.3 Impact of Control Packet Period and Threshold Level

Optimal values for the update period and threshold level was also determined through testing. The values on learning rate and discount factor was taken as the average best performing configuration among the percentage density of coin-cell devices at the specific number of devices, see Section 6.2 and Table 6.3 column five. The impact of different values for period and threshold for the different percent levels of coin-cell devices are displayed in the Figure B.1 for 35 devices, B.2 for 55 devices and B.3 for 75 devices, found in Appendix B. The combined results for the average best configuration are seen in Figure 6.1 Graphs 6.1b, 6.1d and 6.1f. The analysis for finding the values was done by taking the average network lifetime of five deployments with one traffic model on each deployment. With 35 devices, the best value for period and threshold are 70 and 50. For 55 devices the optimal value for the period are 30 with a threshold on 50. For 75 devices the values for the period and the threshold were both much lower, with the same value of 10 each. All the results for the number of devices and individual percent levels of coin-cell devices, are summarized in Table 6.4.

<table>
<thead>
<tr>
<th>Devices</th>
<th>30%</th>
<th>50%</th>
<th>70%</th>
<th>Average</th>
</tr>
</thead>
<tbody>
<tr>
<td>35</td>
<td>α = 0.2, ω = 0.7</td>
<td>α = 0.2, ω = 0.8</td>
<td>α = 0.1, ω = 1.0</td>
<td>α = 0.1, ω = 0.8</td>
</tr>
<tr>
<td>55</td>
<td>α = 0.1, ω = 0.9</td>
<td>α = 0.1, ω = 0.9</td>
<td>α = 0.2, ω = 1.0</td>
<td>α = 0.1, ω = 0.9</td>
</tr>
<tr>
<td>75</td>
<td>α = 0.2, ω = 0.8</td>
<td>α = 0.2, ω = 0.9</td>
<td>α = 0.2, ω = 0.9</td>
<td>α = 0.2, ω = 0.9</td>
</tr>
</tbody>
</table>

Table 6.3: Summary of the best performing parameter configuration for every number of device with each coin-cell density, found in Section 6.2.
Table 6.4: Summary of the best performing parameter configuration for every number of device with each coin-cell density, found in Section 6.3.

<table>
<thead>
<tr>
<th>Devices</th>
<th>30%</th>
<th>50%</th>
<th>70%</th>
<th>Average</th>
</tr>
</thead>
<tbody>
<tr>
<td>35</td>
<td>Threshold = 50, Period = 40</td>
<td>Threshold = 50, Period = 110</td>
<td>Threshold = 50, Period = 70</td>
<td>Threshold = 50, Period = 70</td>
</tr>
<tr>
<td>55</td>
<td>Threshold = 70, Period = 50</td>
<td>Threshold = 100, Period = 130</td>
<td>Threshold = 50, Period = 130</td>
<td>Threshold = 50, Period = 30</td>
</tr>
<tr>
<td>75</td>
<td>Threshold = 10, Period = 10</td>
<td>Threshold = 20, Period = 30</td>
<td>Threshold = 10, Period = 10</td>
<td>Threshold = 10, Period = 10</td>
</tr>
</tbody>
</table>

6.4 Simulation Result

In this section the result from every simulation will be presented, graphs can both be found in this chapter and in the Appendix A, B and C. The configuration on learning rate, discount factor weight, update period and threshold will be set from best average values from Section 6.2 and 6.3. The parameters for individual number of devices and percent levels concluded in Section 6.2 and 6.3 can be seen in a summary in Table 6.3 and 6.4. Where each parameter configuration equals the best performance in network lifetime for that structure on the network.

6.4.1 35 Devices

In Figure C.1 the result from 35 devices are presented in Graph C.1a, C.1b and C.1c for individual coin-cell percent levels of 30%, 50% and 70%. The average node degree for 35 devices is 2.21. The Q-learning routing algorithm gets a performance decrease compared to EAR on all of the three percentage levels of coin-cell devices. With a density of 30% coin-cell the decrease is of 19%, 50% coin-cell got a 21% decrease and with 35 devices and 70% coin-cell the decrease is 16% in network lifetime, thus in performance. The average performance increase/decrease is of 19% decrease. Comparing Q-learning routing to SP routing there is a performance increase/decrease of 1%, -1% and 4% respectively with an overall average performance increase of 1%.

6.4.2 55 Devices

The result from 55 devices are in Figure C.1 Graph C.1d, C.1e and C.1f for individual coin-cell percent levels of 30%, 50% and 70%. The average node degree for 55 devices is 2.39. Same as on 35 devices, the Q-learning routing algorithm
Figure 6.1: Overview of the average results for Learning Rate, Discount Factor Weight, Period and Threshold for 35, 55 and 75 devices.
gets a performance decrease when compared to EAR on all the percentage levels. In graph C.1d in Figure C.1 the performance decrease is 12%. Graph C.1e shows the decrease compared to EAR is 14% and in graph C.1f there is a 5% decrease in performance. For average overall percent levels on 55 devices the decrease in performance is 10%. When compared to SP, there is a performance increase of 23%, 19% and 31% for respective percentages and an overall average result of 25%.

6.4.3 75 Devices
The result from 75 devices are in Figure C.1 Graph C.1g, C.1h and C.1i for individual coin-cell percent levels of 30%, 50% and 70%. The average node degree for 75 devices is 2.54. Unlike the result from 35 and 55 devices the Q-learning routing algorithm gets a performance increase when executed on a network with 75 devices compared to EAR. With 30% coin-cell devices the average network lifetime extended from 468 to 898 time units, an increase of 92% and compared to SP the increase is 186%. At 50% coin-cell devices the performance increase is 92% compared to EAR and 194% when compared with SP. Finally, with a density of coin-cell devices at 70% the increase in performance is 189%. Which leads to an average performance increase of 124% at 75 devices.

6.4.4 Use Case
Result for the special use case of 100 devices are shown in Figure C.2. Compared to EAR the average network lifetime extended from 328 to 1473 time units, in percent that is an increase in performance of 349%.
Chapter 7

Conclusions and Future Work

To start with, this chapter discusses our solution and gives an explanation of the design choices made. Furthermore, the research question is answered. Next, positive effects and drawbacks are evaluated. Finally, suggestions for future work are made.

7.1 Discussion and Result Evaluation

Our proposed routing algorithm which are based on RL got a diversified result. When applied on sparse topology it had a hard time competing with the reference algorithm EAR, but it performed better than SP. However, when the density of devices in the topology increased, our Q-learning algorithm managed to outperform both of the reference algorithms in network lifetime. As said before, both SP and EAR are static routing algorithms, thus they do not change the selected routes during the simulation. Our Q-learning approach is distributed and dynamic, and each node only knows information about its next-hop neighbors. This makes the learning phase of the network with the Q-learning routing algorithm more expensive compared to the flooding technique used by SP or EAR. Also, since it is not centralized the cost for sending characteristic information between nodes are a drawback compared to the reference algorithms were this information is shared among the nodes at a free cost. But because of our implementation of the threshold and update period the impact of this drawback is decreased but not eliminated. With the ability to change routes as time passes, constantly updating the knowledge about the network is essential to make optimal choices for routing. There is a trade-off between how often these values should be updated and the cost of sending current information between nodes. Updating too frequently the benefit of having the latest information vanishes. The opposite, update these values too rarely makes it harder to select the optimal routes because of the outdated information. When
considering a network with a lower average node degree, the possible routes in the network are less compared to a network with a higher average node degree. The positive effect of dynamically changing routes is more significant when the node degree is higher and more routes are available. In a denser network the load on each critical node is higher, thus directing the traffic in the network and balance it can reduce the load on the network and increasing the network lifetime. Our proposed routing algorithm are more resistant to high load than the two reference algorithms. That is how a distributed RL routing algorithm can positively affect the lifetime of a BLE mesh network constructed with heterogeneous devices. By making smarter choices in the network and allowing itself to change the routes dynamically as time goes.

7.2 Future Work

Our study was limited by time. Other limits were caused by the given simulation environment, it was made for a static network with the function to measure the network lifetime of a BLE mesh network. Further, this motive the following proposed future work.

- Apply different Q-value approximation functions. Our proposed Q-value approximation has not been tested against other similar ones in this simulation environment. Valid future work would therefore be to tweak and test other Q-value approximation functions, to find the optimal one for the specific case our study was based on.

- Apply the algorithm in dynamically changing BLE mesh networks. The network could change in different ways. For example, the devices could be constantly moving. More, the generated traffic from each device could vary from low to high load. All these factors would put new demands on the algorithm.

- Add or change parameters in the discount factor. Our study was focused on the network lifetime. Considering other performance metric, other parameters in the discount factor could be of interest. Estimated delivery time to destination, mobility factor or other kinds of device characteristic are possible suggestions.
Bibliography


Appendix A

Impact of Learning Rate and Discount Factor Weight
Figure A.1: 35 Devices, Learning Rate and Discount Factor Weight; 30%, 50%, 70% and average.
(a) 55 Devices, 30% coin-cell.
(b) 55 Devices, 50% coin-cell.
(c) 55 Devices, 70% coin-cell.
(d) 55 Devices, average.

Figure A.2: 55 Devices, Learning Rate and Discount Factor Weight; 30%, 50%, 70% and average.
Figure A.3: 75 Devices, Learning Rate and Discount Factor Weight; 30%, 50%, 70% and average.
Appendix B

Impact of Period and Threshold
Figure B.1: 35 Devices, Period and Threshold; 30%, 50%, 70% and average.
Figure B.2: 55 Devices, Period and Threshold; 30%, 50%, 70% and average.
Figure B.3: 75 Devices, Period and Threshold; 30%, 50%, 70% and average.
Appendix C

Impact of Routing Algorithms
Figure C.1: Comparison of different Routing Algorithms on 35, 55 and 75 devices. The Average Node Degree and coin-cell percentage is displayed on each topology.
Figure C.2: Use case, 100 Devices, 34% coin-cell