Mälardalen University Press Licentiate Theses
No. 319

Mobility Support in Fog-Assisted IoT Networks

Iliar Rabet

March 2022

School of Innovation, Design and Engineering

Abstract

The emerging applications of the Internet of Things (IoT) require more resiliency against dynamism and frequent topological changes in the networks. This is evident in the case of mobile nodes that are penetrating many aspects of human life and thus obsoleting the legacy architecture of IoT systems and challenging the network designers to meet the connectivity requirements. In parallel, enabling technologies such as Software Defined Networking (SDN) and Fog computing facilitate the transition to reliable wireless connectivity. A central controller that resides in an external Fog device can lift the burden of resource-hungry mobility solutions from constrained IoT devices and assist the distributed process of routing and network management. This method however comes with an inherent control overhead that increases with the dynamism in the network.

The state-of-the-art mobility solutions that are proposed for the IETF protocol stack for IoT are limited to constrained devices and thus tend to simplify the processing of the information. The first contribution of this Thesis introduces the MobiFog framework which develops lightweight algorithms to perform seamless handoff and tune network parameters for the new Fog-assisted architecture. The second contribution presents the SDMob framework which focuses on employing sophisticated tracking algorithms (such as particle filter) in the edge devices. In the last contribution of this Thesis, we implement and integrate a standard-compliant protocol extension to Routing Protocol for Low-power and lossy networks (RPL), the de-facto routing protocol for IoT networks. The protocol is called RPL with Route Projection (RPL-RP) and aims at improving the performance of RPL in terms of upstream and east-west traffic by employing a central controller.

A number of factors are known to affect the reliability of a mobility management solution. Some of the main parameters are frequency of data/control message exchanges, the traffic pattern (upward, downward or east-west traffic or probability of sending packets in batches), the mobility pattern (including velocity, sharp turns, probability of being in blind spots), and the environmen-

tal characteristics for wireless propagation (available spectrum range, interference, multi-path shadows). This Thesis considers analytical modeling, simulation, and experiments to evaluate the impact of the relevant parameters on the handoff performance in terms of packet delivery ratio and handoff latency.

Our results provide compelling evidence that all three contributions improved the Quality of Service in various scenarios. The proposed Fog-assisted mobility solution (SDMob) improves the reliability of mRPL (a non-Fog-assisted mobility solution) and ARMOR (a state-of-the-art distributed mobility solution) for a limited number of mobile nodes. MobiFog was evaluated using analytical modeling and was shown to improve the handoff delay with a high probability of seamless handoff. For RPL-RP we achieved enhanced packet delivery ratio and lower delay for east-west traffic due to the improved routing by employing a central controller.

Sammanfattning

De framväxande tillämpningarna av Internet of Things (IoT) kräver bättre anpassning till dynamiska och täta topologiska förändringar i nätverken. Detta är uppenbart i fallet med mobila noder, vilka används i många aspekter av mänskligt liv, och detta gör IoT-systemens äldre arkitektur föråldrad. Parallellt har nya teknologier som Software Defined Networking och Fog computing dykt upp. Dessa underlättar övergången till mer pålitlig trådlös anslutning. En centraliserad styrenhet i den externa Fog-enheten kan överta bördan att exekvera resurshungriga algoritmer från de begränsade IoT-enheterna, och kan hjälpa distribuerade processer med routing och nätverkshantering. Denna metod erbjuder ett intelligent sätt att förutsäga mobilnodens position och dess länkkvaliteter. Existerande mobilitetslösningar är begränsade i sitt stöd av mobilitet och informationsbehandling. Ett av bidragen i denna avhandling är att fylla detta tomrum och utveckla lämpliga algoritmer för att förutsäga handoff- och nätverksparametrar för den nya Fog-understödda arkitekturen. Det andra bidraget i denna licentiatavhandling är ett protokoll genom vilket den centraliserade styrenheten blir involverad i den distribuerade routing-/protokollstacken. Vi har implementerat och integrerat ett standardkompatibelt protokolltillägg till Routing Protocol for Low-power and lossy networks (RPL), det existerande de-facto routingprotokollet för IoT-nätverk.

Ett antal faktorer är kända för att påverka tillförlitligheten hos en mobilitetshanteringslösning. Bland dessa faktorer finns data- och kontrollpaketens frekvens, trafikmönstret (uppåt, nedåt eller öst-västlig trafik, eller sannolikheten att skicka paket i omgångar), mobilitetsmönstret (inklusive hastighet, skarpa svängar, sannolikhet att vara i döda fläckar) samt miljöegenskaperna för den trådlösa överföringen (tillgängligt spektrumområde, interferens, flervägsskuggor). Den här avhandlingen presenterar analytisk modellering, simulering och experiment för att utvärdera inverkan av dessa parametrar på prestandamått såsom andelen levererade paket och handoff-fördröjning.

Den föreslagna Fog-assisterade mobilitetslösningen (SDMob-modell) för-

bättrar tillförlitligheten för mRPL när det gäller andelen levererade paket för en enstaka mobil nod. Våra resultat visar också att SDMob överträffar en modern mobilitetslösning kallad ARMOR för upp till 3 mobila noder. I RPL-RP-modellen, en lösning för punkt-till-punkt-trafik, uppnår vi en förbättrad andel levererade paket och lägre fördröjning, på grund av den förbättrade routingen genom att använda en centraliserad styrenhet.

"To my better half, Rozhan"

Acknowledgement

Throughout the writing of this dissertation I have received a great deal of support and assistance.

I would first like to thank my supervisors Mats Björkman, Maryam Vahabi and Mário Alves. I would particularly like to single out Hossein Fotouhi, whose expertise and day to day guidance was invaluable in formulating this thesis. I'm proud of, and grateful for, my time working with Hossein. Also thanks to my committee members, Carlo Alberto Boano, Oana Iova, José Ramiro Martínez de Dios and Moris Behnam. And a special thanks goes to Mohammad Ashjaei and Sasikumar Punnekkat for reviewing this Thesis.

I would like to acknowledge my friends and colleagues at MDU for their wise counsel and wonderful time we had together. Ali Zoljoodi, Mohammad Loni, Ramon, Priyan, Ali Balador, Mohammad Hassan Adeli, Zeinab Bakhshi, Bahar Houtan, Fereidoun Moradi, Zenepe Satka, Aldin Berisa and all the other amazing people whom I was lucky to know but cannot fit their names in here.

Last but not least, I could not have completed this dissertation without the support of my dear family. Rozhan, my wonderful partner, you were always there for me. Mom, dad and Rahmat Thank you for giving all your kind support and sympathetic ear.

Iliar Rabet
Feb 2022

List of Publications

Papers included in this thesis[1]

Paper A
Title. MobiFog: Mobility Management Framework for Fog-assisted IoT Networks
Venue. IEEE Global Conference on Internet of Things (GCIoT)
United Arab Emirates, Dubai, December 4th - 7th, 2019
Authors. Hossein Fotouhi, Maryam Vahabi, Iliar Rabet, Mats Björkman, and Mário Alves

Paper B
Title. Poster: Particle Filter for Handoff Prediction in SDN-based IoT Networks
Venue. International Conference on Embedded Wireless Systems and Networks (EWSN)
February 17-19, 2020, Lyon, France
Authors. Iliar Rabet, Shunmuga Priyan Selvaraju, Hossein Fotouhi, Maryam Vahabi and Mats Björkman

Paper C
Title. Pushing IoT Mobility Management to *the Edge*: Granting RPL Accurate Localization and Routing
Venue. IEEE 7th World Forum on Internet of Things, 14 June-31 July 2021, New Orleans, Louisiana, USA
Authors. Iliar Rabet, Shunmuga Priyan Selvaraju, Mohammad Hassan Adeli, Hossein Fotouhi, Ali Balador, Maryam Vahabi, Mário Alves, Mats Björkman

[1]The included papers have been reformatted to comply with the thesis layout.

Paper D
Title. RPL-RP: RPL with Route Projection for Transversal Routing
Venue. IEEE 7th World Forum on Internet of Things
14 June-31 July 2021, New Orleans, Louisiana, USA
Authors. Iliar Rabet, Hossein Fotouhi, Maryam Vahabi, Mário Alves, Mats Björkman

Paper E
Title. SDMob: SDN-based Mobility Management for Fog-assisted IoT Networks
Venue. Journal of Sensor and Actuator Networks, Year 2022, Volume 11, Number 1
Publisher. Multidisciplinary Digital Publishing Institute
Authors. Iliar Rabet, Shunmuga Priyan Selvaraju, Hossein Fotouhi, Mário Alves, Maryam Vahabi, Ali Balador, Mats Björkman

Contents

Part I

Thesis

Chapter 1

Introduction

With the rapid development of the Internet of Things (IoT), many emerging and future applications will require or at least benefit from more heterogeneous devices (different hardware/software platforms) that may (or must) physically move. This means that IoT networks must tolerate more topological changes [86]. Frequent changes in the topology, if not handled correctly, can result in the loss of important sensing data or imposing large delays for time-sensitive applications such as healthcare monitoring and factory automation.

Low-power and Lossy Networks (LLN) are one of the prominent communication technologies in the IoT domain [74]. LLNs are a class of networks in which (i) nodes are constrained in terms of energy, memory, and computational resources, (ii) links are lossy and time-varying, and (iii) the deployed topologies can consist of a dense or sparse mesh of routers [50].

Nevertheless, the *de-facto* protocol stack (Table 1.1) for LLNs has not been designed for coping with a dynamic topology. This protocol stack consists of Routing Protocol for Low-Power and Lossy Networks (RPL) [85] for routing, IPv6 over Low Power Personal Area Networks (6LoWPAN), as a low-power alternative to IPv6, on top of a low-power radio standardized as IEEE 802.15.4. The main goal of these protocols is to provide IP-compatible connectivity (mostly collection-based traffic) for a scalable network of battery-powered embedded wireless devices, communicating over multiple hops, that usually run constrained operating systems such as Contiki [15] or TinyOS [44]. Considering the above mentioned challenges, these nodes cannot provide a timely and accurate response to constant and fast topological changes in the network, even for a limited number of mobile nodes.

Table 1.1: The common protocol stack for Low-power and Lossy Networks.

Application	CoAP
Transport	UDP
Routing	RPL
IP	6LoWPAN
MAC	IEEE 802.15.4 MAC
Physical	IEEE 802.15.4 PHY

1.1 Challenges in supporting mobile nodes in LLNs

Mobility management algorithms and protocols are required to improve the network Quality-of-Service (QoS) in terms of reliability, timeliness, availability, and energy consumption, upon changes in the network topology resulting from nodes' mobility. The network should be able to seamlessly and dynamically adapt to these changes, eventually maintaining redundant backup routes to guarantee a smooth network operation.

Mobility support can be integrated with all of the mentioned layers. There exists a substantial body of research regarding MAC layer extensions for different modes of IEEE 802.15.4 such as Time Slotted Channel Hopping (TSCH) that focus on better connectivity for mobile nodes [1, 18, 54, 76]. There is also a neighbor discovery mechanism in 6LoWPAN to support mobile nodes [60]. However, routing protocols are known to be more effective in handling mobile nodes as they have the possibility to change the topology instead of improving a specific lossy link. The focus of this Licentiate research is on supporting mobility in the routing layer (RPL) of low-power IoT network stack, where Fog computing acts as an enabler for data processing algorithms in order to improve network characteristics.

It has been proven that he baseline RPL protocol shows degraded QoS upon mobility [42]. It has been shown that different mobility patterns affect the behavior of RPL in distinct ways [70]. In the standard RPL, the Trickle algorithm is responsible for adapting the transmission frequency of control packets to the rate of changes in the topology. These topological changes can be due to mobility, flipping links, obstacles, or hardware flaws. If nodes do not detect any changes in the topology, they double the interval between control packets, in order to save precious battery lifetime. In case there is any change in the topology, the interval drops to its minimum to handle network dynamics, and link quality changes [43]. This leads to an important trade-off that increasing the control packet rate could result in better resiliency against mobility but at the cost of higher communication overheads and energy expenditure. Nevertheless, since no predictive measure is taken, mobile node routes are only updated after disconnection, leading to network inaccessibility periods that will cause packet loss or delay.

A proactive approach to support seamless handoff could rely on different predictive techniques such as Bayesian filters (e.g. Kalman filters) or Machine Learning models to forecast the future position and link quality of the mobile nodes. Here, the term "filter" is referring to the methods that estimate the state of a temporal variable, which is usually observed under noisy measurements [72].

It is common to have *a priori* knowledge of the number of static nodes (in fixed positions) and the mobile nodes being equipped by an *Inertial Measurement Unit* (IMU) or other sensors. In such a scenario, it is practical to exploit statistical Bayesian prediction models (like Kalman and Particle filters) to fuse position metrics with wireless link quality metrics such as Received Signal Strength Indicator (RSSI). Fusion of information sourced from different physical phenomena such as radio and gyroscope, the mobile nodes benefit from an accurate localization that leads to improvement in network responsiveness.

A typical mobility solution is to implement a handoff mechanism, where mobile nodes are supposed to switch from one point of attachment to another; these points of attachment are usually known as Access Points (APs) and the network of APs is so-called fixed infrastructure. Conventional handoff mechanisms have been designed mostly in a distributed manner, where mobile nodes are assumed to perform 100% of handoff duties [20, 21, 25].

Implementing accurate predictive models may require higher computation capacity than mainstream IoT devices can afford. Resource-constrained nodes can hardly support lightweight filters (such as Kalman filters), provided that the position of the static nodes has to be hardcoded in the mobile node. These limitations can be alleviated by offloading the computation burden to some external entity, such as a Software-Defined Networking (SDN) controller. Employing such a controller, besides its advantages, can introduce new challenges since the extra control packets between the SDN controller and the nodes may lead to overloading of the network. Using an SDN controller for mobility management will require redesign of the handoff mechanism. In this Thesis, we design an SDN-based mobility management architecture that relies on simple yet accurate localization mechanisms running in the SDN controller.

1.2 Challenges in designing predictive models.

In a simplistic approach, the relationship between distance and link quality components can be modeled using the path loss equation. According to Equation 1.1, RSSI at distance D compared to a base distance D_0 drops with a logarithmic relation, where α is a constant value that depends on the environment. This simple model of radio propagation is subject to error sourced by interference, multi-path fading, obstacles, and even changes in the temperature.

$$RSSI_D - RSSI_{D_0} = -10\alpha.log(\frac{D_0}{D}) \tag{1.1}$$

Kalman filters are shown to be unbiased (the average error across all the recursive runs is zero), consistent (the filter is neither overconfident nor under-confident) and optimal (minimum estimation error) [72]. However, in a Kalman filter, the posterior distribution (after the observations) can be computed in closed form only when the relationship between states and observations is linear and the measurement and prediction noise follows a Gaussian distribution [11]. To address nonlinear system models, *Extended Kalman Filters* (EKF) may be preferred; they use *Taylor series* to linearize the equations, trading for a negligible approximation error. On the other hand, Unscented Kalman Filter (UKF) and Particle filters have shown higher accuracy in prediction with a bi-modal distribution of the error [72]. Table 1.2 summarizes some of the characteristics of the above-mentioned filters.

Table 1.2: Comparison of basic filtering methods.

Filter	Pros	Cons
Kalman Filter	Lightweight implementation Optimal filter: minimum error and unbiased	Limited to linear trajectories Limited to Gaussian noise
EKF/UKR	Lightweight implementation Supports non-linear trajectories	Limited to Gaussian noise
Particle Filter	Supports non-linear and Non-Gaussian models	High overhead

1.3 Challenges in designing an SDN-based IoT architecture.

Besides the resource constraint property of IoT networks, lack of possibility to interact with a centralized network manager can limit the dynamicity, flexibility, and reconfigurability within the network. SDN involves a centralized entity in the distributed operation of the network. This can benefit the network in terms of optimizing the resources but introduces new challenges. Employing a mainstream SDN solutions in a network with constrained devices is subject to major complications. The SDN solution needs to address these limitations:

- Protocol limitations: The conventional south-bound API for SDN, OpenFlow, is not suitable for Low-power and Lossy Networks. The packets are too large (after appending a 6LowPAN header), thus getting fragmented, and their frequency is not tuned for low power consumption. A reasonable design choice is that SDN controller makes minimal changes to the existing routes defined by RPL.

- Architectural limitations: All the nodes in the network may request control packets. The conventional SDN architecture in which the routers ask the controller for each flow is not convenient for a network that may scale to thousands of nodes. The SDN controller needs to rely on the legacy RPL for basic connectivity and optimize its parameters for the sake of scalability.

- Limitations in the Implementation: In conventional SDN, matching the flows can be implemented using specific fields in the header. The number of matching fields and the number of entries in the flow table are also limited on low-power devices.

- Limitations in the controller: The controller may be implemented on one/multiple embedded devices or using an external border router. Either way, controlling a Low Power and Lossy network mandates considering its unique characteristics. If the controller is implemented on an embedded device, it is also subject to severe computation constraints.

There have been many efforts to design SDN controllers for IoT networks, specifically for sensor networks [14, 27, 46]. However, most of these works were quite preliminary, lacking real algorithm design and evaluation in Commercial off-the-shelf (COTS)/standard platforms. Another important aspect

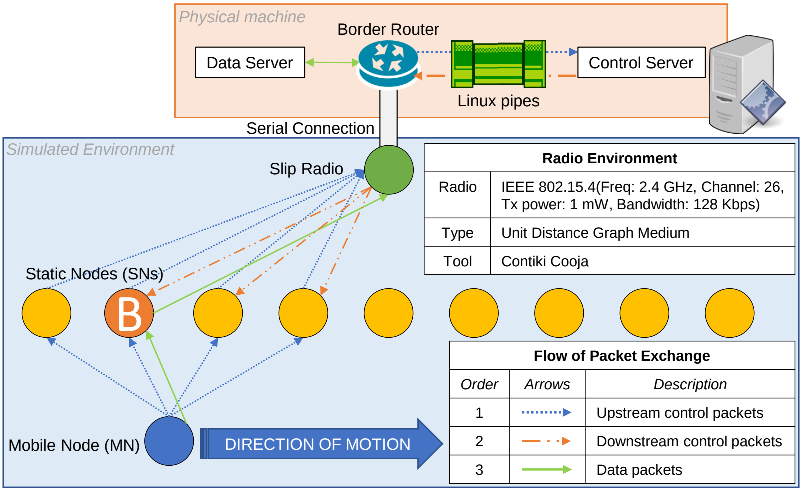


Figure 1.1: Offloading the mobility management to the controller.

that we try to emphasize is using the centralized entity for mobility management. As illustrated in Figure 1.1, before updating the routing states, the controller needs to collect specific information from the mobile node to be able to apply the filtering algorithm.

A recent effort by the IETF ROLL group [79] has also focused on the above-mentioned problems. The draft on the "root initiated routing state in RPL" defines new control packet types and control packet options that provide backward-compatible mechanisms for the RPL root to collect more information and install the so-called *projected routes* on the selected nodes. One of the contributions of this Thesis (RPL-RP) is an SDN-based platform that is inspired by this draft.

1.4 Outline of Thesis

The Thesis is structured as follows: Chapter 2 discusses the previous work on the topic and classifies them according to using a centralized controller and Bayesian filters. Next in chapter 3, the research questions and corresponding contributions are discussed. Chapter 4 discusses the possible future works and concludes the Thesis.

Chapter 2

Background

Mobility support in the IoT has been a topic of interest since mobile devices have penetrated many aspects of our life and heterogeneity in the model of development and deployment of these infrastructures is rapidly increasing. They may have different hardware/software platforms that may move and impose changes to the topology of the network.

There have been many efforts in devising mobility management solutions for IoT and wireless sensor networks. In this section, we review state-of-the-art and identify existing gaps and key research questions to be answered when designing Fog-assisted mobility management. Mobility management can be performed at different layers of the protocol stack. Nevertheless, to avoid routing loops, mobility management should also be considered at the routing layer [36].

RPL is considered as the *de-facto* routing protocol for IoT. While it naturally supports the joining and leaving of nodes, it performs poorly upon the dynamics imposed on the network topology. RPL maintains a distributed data structure named *Destination Oriented Directed Acyclic Graph* (DODAG). The process starts with the root transmitting a DODAG *Information Object* (DIO) that embeds the needed information to construct a routing tree towards the root.

RPL allows two modes of operation –*storing* and *non-storing*– for downstream traffic. In non-storing mode, it is only the root that maintains the downward routes. This mode scales better since the memory footprint at intermediate nodes does not increase with the size of the network. It should be pointed out that in RPL it is more challenging to support mobility for downstream traffic since a mobile node must notify the root (rather than only updating its parent for upstream traffic).

There are some limitations inherent to the design of the RPL that are biased to the upward collection-based traffic and all the other traffic directions are

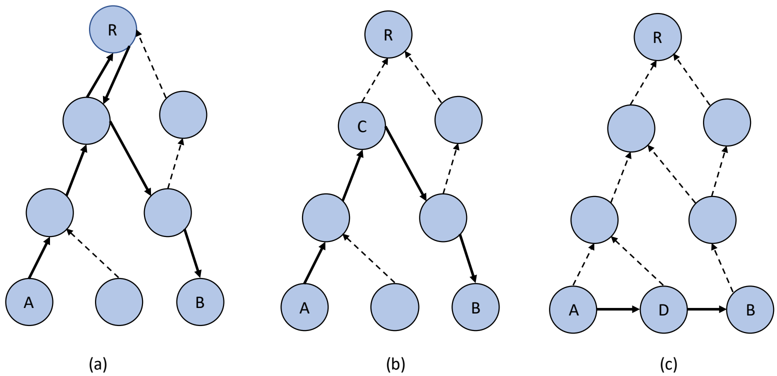


Figure 2.1: Examples of data transmission strategies in (a) RPL's *non-storing* mode (b) RPL's *storing mode* (c) A shorter route including siblings.

restricted to transmit only along the DAG resulting in stretched [1] routes. For downward communication or any-to-any communication, the nodes in RPL are programmed to be either in *Storing* or *Non-Storing* mode, where the former requires more memory for keeping track of the nodes in the sub-trees of each node. For routing between two non-storing nodes in the network, the packets have to be transmitted up to the root and back down to the destination. Even in the storing mode, the source node has to find a common ancestor with the destination, which is not necessarily the optimal path. Figure 2.1 exhibits how different modes of RPL stretch the routes, while siblings can promote point-to-point routing. Besides the routing mode, the common metrics used by the RPL's objective function prefer the routes that have better upward connectivity, while links can show asymmetric behavior in different directions [16].

Integrating the central controller specifically needs reliable downward communication. Furthermore, new applications not only require mobile nodes but also new traffic patterns such as downward and transversal (east-west). So it is of utmost importance to consider different traffic patterns alongside the various mobility patterns.

2.1 Extensions to RPL for mobility support

The authors in [36] classify RPL enhancements to support mobility into scenarios with networks including only mobile nodes (e.g. VANETs) and networks

[1] the routes being longer because of being limited to only use parents to reach any node

with both static and mobile nodes. For the former, the recommendations of the RPL standard to not set the mobile nodes as routers cannot be respected. In this case, Tian et. al [83] try to adjust the Trickle timer according to mobile nodes' velocity and utilize geographical information as an RPL metric. In case the mobile node is not equipped with IMU sensors, it is possible to estimate its position through the *Doppler Effect*, as explained in [60].

Smart-HOP [25] has two main phases of Data Transmission and Discovery. The Mobile Node (MN) periodically monitors the link quality level by receiving beacons from the serving AP (current parent node). Upon receiving several data packets in a given window, the serving AP replies with the average received signal strength (ARSSI) or average signal to noise ratio (ASNR).

The MN disconnects from the serving AP when the link quality degrades or breaks (no packet reception). Thus, MN immediately enters the Discovery Phase by broadcasting burst of beacons to neighbor APs, while expecting replies after each burst. When a high link quality (ARSSI greater than a threshold level) is detected, the MN attaches to the new serving AP.

mRPL [20] is the smart-HOP algorithm integrated within the standard RPL routing. The major changes are: (i) the use of RPL control messages instead of beacons (i.e. DIS and DIO messages), and (ii) the additional timers to increase handoff efficiency and reliability. The process of dropping, keeping, and assessing link(s) is similar to the smart-HOP mechanism.

Unlike smart-HOP, the MN keeps data communication with the serving AP until it finds a better AP. After a successful handoff, the nullifying process of the RPL algorithm is executed.

mRPL+ [21] is a combination of the hard and soft handoff approaches within the RPL routing. The main contribution of mPL+ is the introduction of the overhearing mechanism. The APs overhear the link (between MN and the serving AP) activities by measuring the average RSSI of the packets received from the MN. The common chipsets such as Chipcon CC2420 MAC sub-layer supports promiscuous mode that sniffs packets.

Handoff in mRPL+ is classified into two main categories: hard and soft handoffs. In a hard handoff, MN disconnects from one AP and searches for a new AP, which implies a disconnection period. Consequently, a hard handoff mechanism is prone to high packet losses. In a soft handoff, the new potential AP is selected, while communicating with the current AP.

Several other works focused on mobility support in commodity IoT protocols by enhancing 6loWPAN, RPL, and CoAP [5, 26, 32, 38]. A recent survey paper collects some of the related works in this area [35]. The main issue of these works is the special focus on one of the network layers, and the lack of ability to re-adapt to architectural changes.

2.2 Software Defined Networking paradigm in Low-Power Networks

There is growing popularity in using SDN-enabled solutions in IoT networks. It is too expensive in terms of energy efficiency and communication overhead to simply integrate the common SDN solutions and standards within constrained IoT networks without re-designing the SDN to consider IoT limitations [41]. Therefore, there is a requirement for devising solutions targeting IoT networks with reduced complexity and operational cost.

A substantial body of knowledge belongs to centralized network management in a constrained environment. Sensor OpenFlow [46] is one of the earliest efforts for offloading the network control to bring about more flexibility against dynamic policies and ease of management. In a similar approach, SDN-WISE [27] defines its own Topology Discovery layer in the constrained nodes to enable cross-layer operations. An extension to SDN-WISE with [9] controller, allows MAC layer scheduling in TSCH mode of IEEE 802.15.4 and increases connectivity for mobile nodes [8].

In μSDN [4], the authors argue that a centralized controller needs to be compatible with legacy RPL nodes and introduce some optimizations that allow μSDN to overcome the constraints that are common to IoT nodes. Optimizations can include avoiding packet fragmentation, source-routed control packets, and configuring update timers. Hydro [13] was another effort to improve the distributed DAG formation with centralized primitives. The main limitation of these works is not being compatible with COTS/standard platforms.

Thubert et. al [81] proposes a framework for an SDN-based TSCH scheduler aiming to meet the requirements of deterministic networking. The authors claimed that the key to improving reliability and mitigating interference/dynamism in the network is diversity. Diversity can be achieved in different domains as spatial diversity is leveraged with multi-path routing, temporal diversity by re-transmissions, and frequency diversity using channel hopping.

The IETF draft on DAO projection [79] defines some primitives to involve the central border router in the distributed operation of RPL and classifies route projection into *Storing Mode Projected Route* (SMPR) and *Non-Storing Mode Projected Routes* (NMPR). The mode for projected routes is independent of RPL's operation mode, meaning that the network can consist of storing mode RPL working with non-storing route projection or vice versa. NMPR uses source routing for the data packets but in SMPR root node asks the source node to update the routing state in all the intermediate nodes. The ROLL working group is currently actively working on this document and to the best of our knowledge, there is not any available implementation to compare with. In both modes of DAO projection, getting acknowledgment from either source or destination would suffice. We suggest a reform to put all the intermediate nodes in direct connection with the controller rather than getting an acknowledgment only from the source or destination of the path. This will ease troubleshooting since the controller gets to know which link in the path is troublesome. The standard however does not specify how and which routes should be calculated.

Efforts have been made (including by standardization bodies) to design solutions for managing IoT networks. The *Internet Engineering Task Force* (IETF) has a recent draft for infusing data routes into the network that is called *DAO projection* [80]. It defines a framework for the root node to initialize some options in DODAG *Advertisement Objects* (DAO) through new control messages, namely *Project DAO Request* (PDR) and *PDR-Acknowledgement* (PDR-ACK). This enables the root node to install routes in either the source or intermediate nodes along the path. The mechanism is a low-overhead substitute for implementing centralized network management in IoT networks.

Except for DAO projection, most of the mentioned works here, do not consider a standard compatible mechanism for interaction with the controller. Unfortunately, DAO projection has not been implemented in the community and one of the contributions of this Thesis is to implement a simplistic profile of this newly proposed draft that considers nodes' mobility.

2.3 Location prediction models for proactive handoff

A major research line focuses on the applicability of localization and tracking algorithms for pro-active mobility management.

The most suitable algorithm to predict the handoff depends on the available sources of online and offline information. For instance, a *Deep Learning*-based Long Short Term Memory (LSTM) model could be preferred, if a supervised data set exists for the model to learn from [87]. Although there are some works such as Co-RPL [26], that define semi-distributed zones around static nodes to localize the mobile nodes. Still, Bayesian Filters, such as Kalman and Particle filter are the classic and most studied tools to track the position of a mobile node in time.

Bayesian filters, usually model the position (state) and link quality and velocity (measurements) as a Hidden Markov Model illustrated in Figure 2.2. This model assumes the Markovian property between the states, meaning that the current state only depends on the previous state. At time step t, the filtering algorithms use the measurement vector y_k to estimate the state vector x_k and predict the future state $x_k + 1$.

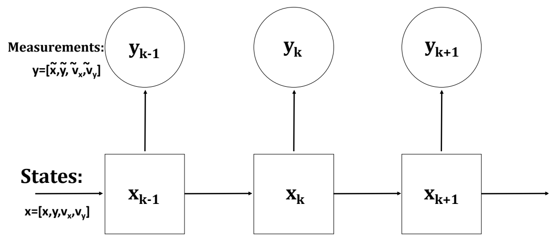


Figure 2.2: Hidden Markov Model for Position (state) and Measurements (RSSI and IMU).

In [6], authors have proposed Kalman RPL, in which a mobile node transmits a beacon that includes its velocity information in specific intervals. After a positioning phase that estimates the current position of the mobile node using three static nodes in its vicinity, it can predict the future position of the node.

EKF-RPL [10] takes a similar approach but it employs EKF within RPL to support non-linear trajectories. There are also some efforts on adopting *on-demand* routing strategies when a node starts searching for a route for transmitting data. The *Lightweight On-Demand Adhoc Distance-vector* routing protocol - Next Generation (LOADng) [12] is one such protocol specifically designed to support any-to-any communication in LLNs, although it is not as well-studied as RPL. EKF-LOADng [30] predicts RSSI after a trilateration positioning. In the triangulation phase, a mobile node broadcasts a message asking for packets from its static neighbors. Responses from static nodes experience a random waiting time to avoid the collision. Another challenge is that the mobile node has to be programmed with the position information of the anchors and perform the sophisticated filter on its own.

Particle filter leads to more accurate results and better resiliency against nonlinear moving trajectory and non-Gaussian noise compared to EKF [72]. Particle filter (a.k.a *Sequential Monte Carlo*) uses hundreds to thousands of samples to predict the future state and fuse the measurement, hence requiring a higher computation capacity than most constrained IoT nodes can provide.

Particle filter has been extensively used to support mobility in Unmanned Aerial Vehicle (UAV) assisted networks [73] or cellular communications as in [48] which also proposes a *Rao-Blackwellised* particle filter as a lightweight alternative to the baseline particle filter. There are fundamental differences between cellular networks and LLNs such as density of the network deployment, range of transmission, and speed of mobile nodes. Authors of [28] also propose to offload the localization to a Fog device that estimates the position of the mobile node by defining zones around the static nodes in a WiFi network.

Although there are some well-known applications for sophisticated and computation-intensive filters/algorithms for network/mobility management in cellular networks, the literature in IoT and low-power networks mostly neglect it. One possible explanation is the lack of convenient SDN-based architectures. Overall, this proposal tries to fill this gap by offloading the resource-hungry algorithms to the more m̈uscledc̈ontroller.

2.4 Reflections on the related works

Reviewing the aforementioned research works we can observe a research gap regarding the implications of employing a centralized controller in the distributed operation of low power networks as none of the previous works considers offloading the Bayesian filters to the controller. Mobility solutions that do not implement a Bayesian filter also have rarely used the centralized architecture due to the communication overhead and challenges that it creates. Hence the factors involved in the performance of a fog-assisted IoT mobility solution have not been thoroughly studied in the literature.

Chapter 3

Thesis contributions

In this section, we specify the research goal of this Thesis, with the aim to achieve seamless mobility management in fog-assisted IoT networks. The contributions listed in this Thesis are designed to support this core goal which is further refined by dividing it into concrete research questions as stated below. The list of published papers has been indexed according to the timing of the publications.

Research Goal: *Support mobility in Fog-assisted IoT networks employing low-power radios while providing QoS requirements.*

This Research Goal is further broken down into specific research questions (RQs).

- **RQ 1**: What are the impacts on network reliability and timeliness of offloading the mobility solution to Fog devices by gathering link quality reports and installing root-initiated routes?

  Motivation: Existing distributed mobility solutions in the literature have many limitations when it comes to the required resources. The resource-constrained IoT devices usually run on a battery and lack a global view of the link qualities across the network. Fog devices have adequate computational capacity to maintain a global view and act based on that. RPL nodes are designed to free up their memory given the constraints they inherit.

- **RQ 2**: Can the new fog-based computing resources provide accurate and timely position predictions by employing computation-intensive filters?

  Motivation: Besides a case that the mobility trajectory of a mobile node is known to a centralized controller, many RPL extensions use Bayesian filters in a distributed manner to track the mobile node and perform predictive routing. The accuracy of the filtering algorithms can dramatically be enhanced by using Fog computing technologies.

- **RQ 3**: What are the communication overheads and benefits of interaction between the centralized controller and the IoT devices if they are compatible with the standard RPL?

  Motivation: Addressing the previous research questions assumed that a mechanism for installing a route in the multi-hop network was agreed upon. Similar to mobility, the protocol stack is not designed to support ease of interaction with a centralized controller. The conventional centralized controllers and the protocols used for manipulating the in-network nodes including OpenFlow are not well-tuned for low-power and resource-constrained networks. Within the framework of this research question, the main goal is to adapt the protocols to reduce the overhead of these protocols and integrate them with standard LLN protocol stack.

- **RQ 4**: Can employing a centralized controller in an IoT network also support more flexible traffic patterns such as downstream (from the edge to the sensors) or transversal traffic (point-to-point traffic among the low power devices)?

  Motivation: As mentioned earlier in Section 2, RPL favors the collection-based, or upstream, traffic, since it was designed for wireless sensor networks. Conventional mobility solutions inherit this characteristic from standard RPL. But, with emerging SDN-based architectures in IoT, there is a demand and also a potential to improve the latency and reliability of the downstream and transversal traffic patterns.

This section provides a summary of the contributions of the Thesis. These contributions have led to the associated publications. In Table 3.1, the relationship between the research goals and the publications is outlined. Later on, we present a summary of each research contribution.

Table 3.1: Mapping of the research questions to the contributions.

	RQ 1	RQ 2	RQ 3	RQ 4
Contribution 1	✓			
Contribution 2	✓	✓		
Contribution 3			✓	✓

3.1 Contribution 1

This research contribution is mostly related to research question 1 and explores the possibility of using the global view at the Fog layer for mobility management, using new timers that facilitate soft and hard handoff mechanisms (not using on filters). Paper A presents MobiFog, a framework that provides the mobile node with improved connectivity. The main idea is that if the current serving parent notices a degrading link, it will notify the controller. In parallel, alternative parents (not the current best parent) also notify the controller in case they can serve the mobile node with a reliable connection. The controller then uses the RSSI reports and defines thresholds to manage the best parent for the mobile nodes in time. (Please note that in this setting the controller is not running a tracking algorithm).

3.1.1 Paper A

Title. MobiFog: Mobility Management Framework for Fog-assisted IoT Networks
Venue. IEEE Global Conference on Internet of Things (GCIoT)
United Arab Emirates, Dubai, December 4th - 7th, 2019
Authors. Hossein Fotouhi, Maryam Vahabi, Iliar Rabet, Mats Björkman, and Mário Alves
Abstract
Mobility is becoming a challenging issue in upcoming IoT applications, where it is crucial to employ mobile entities. In parallel, Fog computing has revolutionized network architecture, while enabling local processing of measurements and reducing bandwidth overhead, which results in a more reliable system and real-time support. However, mobility management is a missing framework within the mobile IoT networks with Fog computing architecture. This paper provides a simple and generic seamless handoff model, dubbed as MobiFog, where it addresses the handoff mechanism with zero delays while providing high reliability.

3.2 Contribution 2

This research contribution addresses the problem of offloading the processing of network control information (particle filter) to the Fog layer and introduces the SDMob frameworks and is associated with the following three publications. Paper B provides an analytical model and paper C evaluates SDMob by conducting simulations with linear and circular mobility patterns and traffic loads (for a single mobile node). Paper E extends the SDMob model for networks with multiple mobile nodes and harsh mobility patterns and compares the analytical results with the simulations.

3.2.1 Paper B

Title. Poster: Particle Filter for Handoff Prediction in SDN-based IoT Networks
Venue. International Conference on Embedded Wireless Systems and Networks (EWSN)
February 17-19, 2020, Lyon (France)
Authors. Iliar Rabet, Shunmuga Priyan Selvaraju, Hossein Fotouhi, Maryam Vahabi, and Mats Björkman
Abstract
The standard implementation of the RPL protocol has struggled to limit the impact of mobility on the throughput of the IoT network. The handoff process is of great importance to optimize the trade-off between the control overhead (for maintaining the network topology), and the delay, caused by nodes' mobility. In this work, We have proposed a method for predicting future handoffs through the fusion of RSSI value and Inertial Measurement Unit (IMU) information using particle filter, which is known for accuracy albeit it needs higher computation capacity. The provided analytical model indicates lower network interruption with the proposed method.

3.2.2 Paper C

Title. Pushing IoT Mobility Management to *the Edge*: Granting RPL Accurate Localization and Routing
Venue. IEEE 7th World Forum on Internet of Things, 14 June-31 July 2021, New Orleans, Louisiana, USA
Authors. Iliar Rabet, Shunmuga Priyan Selvaraju, Mohammad Hassan Adeli, Hossein Fotouhi, Ali Balador, Maryam Vahabi, Mário Alves, Mats Björkman
Abstract. Accurate and timely mobility support in the Internet of Things (IoT) applications is a challenging issue, considering the inherent scarce resources of IoT devices. However, the computational, memory, and communication burden may be pushed into more "muscled" *Software Defined Network* (SDN) controllers. A centralized controller can exploit its global view of the network to predict and support seamless handovers.

However, it requires the controller to be enhanced with extra link quality information. In this work, we present SDMob, an SDN-based mobility management solution that lifts the burden of computation-intensive filtering algorithms from resource-constrained nodes and achieves accurate and fast handovers upon nodes' mobility under *Routing Protocol for Lossy Low-power Networks* (RPL) and *IPv6 over Low-Power Wireless Personal Area Networks* (6LoWPAN). We show that SDMob improves the baseline RPL and the selected benchmark mRPL in terms of packet delivery ratio leveraging more reliable routing and applying *Particle* filter and variations of *Kalman* filter on radio signal strength data enables more accurate localization for complex real-world trajectories.

3.2.3 Paper E

Title. SDMob: SDN based Mobility Management for Fog-assisted IoT Networks

Authors. Iliar Rabet , Shunmuga Priyan Selvaraju, Hossein Fotouhi, Mário Alves, Maryam Vahabi, Ali Balador, Mats Björkman

Abstract Internet-of-Things (IoT) applications are envisaged to evolve to support the mobility of devices while providing quality of service in the system. To keep the connectivity of the constrained nodes upon topological changes, it is of vital importance to enhance the standard protocol stack, including *Routing Protocol for Lossy Low-power Networks* (RPL), with accurate and real-time control decisions. We argue that devising a centralized mobility management solution based on a lightweight Software Defined Networking (SDN) controller provides seamless handoff with reasonable communication overhead. A centralized controller can exploit its global view of the network, computation capacity, and flexibility to predict and significantly improve the responsiveness of the network. This approach requires the controller to be fed with the required input and to get involved in the distributed operation of the standard RPL. We present an extension to our previous work *SDMob* [63], which is a lightweight SDN-based mobility management architecture that integrates an external controller with a standard-compatible constrained network and lifts the burden of computation-intensive filtering algorithms away from the resource-constrained nodes to achieve seamless handoffs upon nodes' mobility. Through extensive analytical modeling and simulations, we show that SDMob outperforms the baseline RPL and the state-of-the-art ARMOR in terms of packet delivery ratio and end-to-end delay, with an adjustable and tolerable overhead. With SDMob the network provides close to 100% packet delivery

ratio (PDR) for a limited number of mobile nodes and maintains sub-meter accuracy in localization under random mobility patterns and varying network topologies.
Note: This recently submitted paper extends paper C for multiple mobile nodes and harsh mobility patterns. SDMob outperforms state-of-the-art ARMOR in packet delivery ratio.

3.3 Contribution 3

This contribution focuses on implementing and evaluatiing a standard-compatible SDN platform (RQ3). This platform, called RPL-RP, can benefit the system in several ways. One of the benefits is to optimize the transveral routing (RQ4). Paper D implements and evaluates the solution using the well-known Contiki-NG/Cooja simulator.

3.3.1 Paper D

Title. RPL-RP: RPL with Route Projection for Transversal Routing
Venue. IEEE 7th World Forum on Internet of Things
14 June-31 July 2021, New Orleans, Louisiana, USA
Authors. Iliar Rabet, Hossein Fotouhi, Maryam Vahabi, Mário Alves, Mats Björkman
Abstract
Routing Protocol for Low-Power and Lossy Networks (RPL) as the most widely used routing protocol for constrained Internet of Things (IoT) devices optimizes the number of routing states that nodes maintain to minimize resource consumption. Given that the routes are optimized for data collection, this leads to selecting sub-optimal routes, particularly in the case of east-west or "transversal" traffic. Additionally, RPL neglects interactions with a central entity in the network for monitoring or managing routes and enabling more flexibility and responsiveness to the system.

In this paper, we present *RPL with Route Projection (RPL-RP)* that enables collecting siblings' relations at the root node in order to inject routing states to the routers. This backward-compatible RPL extension still favors collection-based traffic patterns but it enriches the way routing protocol handles other flow directions. We address different advantages of RPL-RP in contrast to standard RPL and evaluate its overhead and improvements in terms of end-to-end delay, control overhead, and packet delivery ratio. Overall, RPL-RP halves the end-to-end delay and increases network reliability by 5%

while increasing network overhead by only 3%.

Chapter 4

Summary and Future Works

With the contributions being discussed, we now turn our attention to conclusions and possible directions for future works.

4.1 Conclusions

We have proposed contributions to assist low-power networks by adding a centralised controller and focusing on applications that include mobile nodes. The implications of integrating such a controller in the distributed operation in the low-power domain have been studied.

One of the main contributions of this Thesis was introducing SDMob, which is an architecture for offloading the mobility solution to a resource-rich Fog device. Delivering the new control traffic upstreams and downstreams becomes challenging in some conditions, and we have proposed mechanisms to overcome these challenges. Our analytical results were confirmed by simulations and show that for a reasonable number of mobile nodes it is beneficial to employ SDMob, since it can support dense networks, harsh mobility patterns, varying hop distance and velocity and path loss variance.

As another contribution, we presented RPL-RP, an extension to RPL that supports injecting point-to-point routes on-demand by a centralized entity. The new system defines new control packet types and options that collect extra sibling information to be visualized in a dashboard. This enables the administrator to define routing states in the in-network nodes. Overall, the evaluation of RPL-RP showed that the improvements incurred by the projected routes can surpass its overheads. Although its performance highly depends on the quality of the projected routes, we showed that with reasonable overhead in control traffic and memory, RPL-RP achieves an almost perfect packet delivery for transversal routes with routes that optimized the latency for hundreds of milliseconds.

MobiFog was another of the contributions of this Thesis, that similar to SDMob, focused on offloading the mobility support to a centralised controller. However, SDMob employed Bayesian filters for predicting the future parents, but MobiFog defines thresholds for RSS values.

4.2 Future work

This section analyses the works that the author is considering for future work after the Licentiate seminar.

- IEEE 802.15.4 has several modes of operation in the MAC layer for different applications. **Time Slotted Channel Hopping** (TSCH) as one of these modes is well-known for the options that suit industrial applications that require reliability. Most of the works regarding mobility define extensions to the routing layer. Studies on the impact of TSCH parameters on mobility support are not mature enough. To mention a few, the TSCH nodes schedule the timeslots and frequencies. The scheduling algorithms are classified as distributed, centralized, and autonomous. The optimal energy consumption is achieved by state-of-the-art "autonomous" schedulers that send zero scheduling packets. A network calculus model will be considered for modeling the service and arrivals.

- There are several communication parameters such as transmission power that are usually hard-coded into constrained nodes. Most of the extensions of the standard protocols do not optimize all the parameters. In a preliminary study, we are analyzing the feasibility of applying Lyapunov optimization [52] to jointly optimize routes and the transmission power.

- The IETF ROLL working group, that is responsible for standardization of RPL, is working on a new draft (since 2016), called **DAO projection** [79]. This draft defines standard-compliant primitives that allow a centralized PCE to project routes into the distributed nodes. We have initiated a collaboration aiming at a full implementation and evaluation of the DAO projection with two of its different profiles that closest match the directions of this Thesis. This work will compare source routing and hop-by-hop projected routes, and their application to mobility support.

Bibliography

[1] Y. Al-Nidawi and A. H. Kemp. Mobility aware framework for timeslotted channel hopping ieee 802.15. 4e sensor networks. *IEEE Sensors Journal*, 15(12):7112–7125, 2015.

[2] E. Ancillotti, C. Vallati, R. Bruno, and E. Mingozzi. A reinforcement learning-based link quality estimation strategy for RPL and its impact on topology management. *Computer Communications*, 112:1–13, 2017.

[3] E. Baccelli, M. Philipp, and M. Goyal. The P2P-RPL routing protocol for IPv6 sensor networks: Testbed experiments. In *SoftCOM 2011, 19th International Conference on Software, Telecommunications and Computer Networks*, pages 1–6. IEEE, 2011.

[4] M. Baddeley, R. Nejabati, G. Oikonomou, M. Sooriyabandara, and D. Simeonidou. Evolving SDN for low-power iot networks. In *2018 4th IEEE Conference on Network Softwarization and Workshops (NetSoft)*, pages 71–79. IEEE, 2018.

[5] G. Bag, M. Raza, K.-H. Kim, and S.-W. Yoo. Lowmob: Intra-pan mobility support schemes for 6lowpan. *Sensors*, 9(7):5844–5877, 2009.

[6] M. Barcelo, A. Correa, J. L. Vicario, A. Morell, and X. Vilajosana. Addressing Mobility in RPL with Position Assisted Metrics. *IEEE Sensors Journal*, 16(7):2151–2161, 2016.

[7] M. Becker, K. Kuladinithi, and C. Görg. Modelling and simulating the trickle algorithm. In *International Conference on Mobile Networks and Management*, pages 135–144. Springer, 2011.

[8] L. L. Bello, A. Lombardo, S. Milardo, G. Patti, and M. Reno. Experimental assessments and analysis of an sdn framework to integrate mobility management in industrial wireless sensor networks. *IEEE Transactions on Industrial Informatics*, 16(8):5586–5595, 2020.

[9] P. Berde, M. Gerola, J. Hart, Y. Higuchi, M. Kobayashi, T. Koide, B. Lantz, B. O'Connor, P. Radoslavov, W. Snow, et al. Onos: towards an open, distributed sdn os. In *Proceedings of the third workshop on Hot topics in software defined networking*, pages 1–6, 2014.

[10] M. Bouaziz, A. Rachedi, and A. Belghith. EKF-MRPL: Advanced mobility support routing protocol for internet of mobile things: Movement

prediction approach. *Future Generation Computer Systems*, 93:822–832, 2019.

[11] O. Cappe, S. J. Godsill, and E. Moulines. An overview of existing methods and recent advances in sequential Monte Carlo. *Proceedings of the IEEE*, 95(5):899–924, 2007.

[12] T. Clausen, J. Yi, and A. C. De Verdiere. LOADng: Towards aodv version 2. In *2012 IEEE Vehicular Technology Conference (VTC Fall)*, pages 1–5. IEEE, 2012.

[13] S. Dawson-Haggerty, A. Tavakoli, and D. Culler. Hydro: A hybrid routing protocol for low-power and lossy networks. In *2010 First IEEE International Conference on Smart Grid Communications*, pages 268–273. IEEE, 2010.

[14] A. De Gante, M. Aslan, and A. Matrawy. Smart wireless sensor network management based on software-defined networking. In *2014 27th Biennial Symposium on Communications (QBSC)*, pages 71–75. IEEE, 2014.

[15] A. Dunkels, O. Schmidt, N. Finne, J. Eriksson, F. Österlind, N. Tsiftes, and M. Durvy. The Contiki OS: The operating system for the internet of things, 2011, 2018.

[16] S. Duquennoy, J. Eriksson, and T. Voigt. Five-nines reliable downward routing in rpl. *arXiv preprint arXiv:1710.02324*, 2017.

[17] S. Duquennoy, O. Landsiedel, and T. Voigt. Let the tree bloom: Scalable opportunistic routing with oRPL. In *Proceedings of the 11th ACM Conference on Embedded Networked Sensor Systems*, pages 1–14, 2013.

[18] A. Elsts, J. Pope, X. Fafoutis, R. J. Piechocki, and G. Oikonomou. Instant: A tsch schedule for data collection from mobile nodes. In *EWSN*, pages 35–46, 2019.

[19] H. Farag, P. Österberg, and M. G. .... RMA-RP: A Reliable Mobility-Aware Routing Protocol for Industrial IoT Networks. In *IEEE Global Conference on Internet of Things (GCIoT)*, 2019.

[20] H. Fotouhi, D. Moreira, and M. Alves. mRPL: Boosting mobility in the Internet of Things. *Ad Hoc Networks*, 26:17–35, 2015.

[21] H. Fotouhi, D. Moreira, M. Alves, and P. Yomsi. mRPL+: A mobility management framework in RPL/6LoWPAN. *Computer Communications*, 104:34–54, 2017.

[22] H. Fotouhi, D. Moreira, M. Alves, and P. Yomsi. mRPL+: A mobility management framework in RPL/6LoWPAN. *Computer Communications*, 104:34–54, 2017.

[23] H. Fotouhi, M. Vahabi, I. Rabet, M. Björkman, and M. Alves. MobiFog: Mobility Management Framework for Fog-assisted IoT Networks. In *IEEE Global Conference on Internet of Things GCIoT'19, 04 Dec 2019, Dubai, United Arab Emirates*, 2019.

[24] H. Fotouhi, M. Vahabi, A. Ray, and M. Björkman. SDN-TAP: an SDN-based traffic aware protocol for wireless sensor networks. In *2016 IEEE 18th International Conference on e-Health Networking, Applications and Services (Healthcom)*, pages 1–6. IEEE, 2016.

[25] H. Fotouhi, M. Zuniga, M. Alves, A. Koubâa, and P. Marrón. Smart-hop: A reliable handoff mechanism for mobile wireless sensor networks. In *European Conference on Wireless Sensor Networks*, pages 131–146. Springer, 2012.

[26] O. Gaddour, A. Koubäa, R. Rangarajan, O. Cheikhrouhou, E. Tovar, and M. Abid. Co-rpl: Rpl routing for mobile low power wireless sensor networks using corona mechanism. In *Proceedings of the 9th IEEE international symposium on industrial embedded systems (SIES 2014)*, pages 200–209. IEEE, 2014.

[27] L. Galluccio, S. Milardo, G. Morabito, and S. Palazzo. SDN-WISE: Design, prototyping and experimentation of a stateful sdn solution for WIreless SEnsor networks. In *2015 IEEE Conference on Computer Communications (INFOCOM)*, pages 513–521. IEEE, 2015.

[28] T. N. Gia, A. M. Rahmani, T. Westerlund, P. Liljeberg, and H. Tenhunen. Fog computing approach for mobility support in internet-of-things systems. *IEEE Access*, 6:36064–36082, 2018.

[29] O. Gnawali and P. Levis. The minimum rank with hysteresis objective function. *RFC 6719*, page 13, 2012.

[30] A. J. Gonçalves, R. A. Rabêlo, J. J. Rodrigues, and L. M. Oliveira. A mobility solution for low power and lossy networks using the LOADng protocol. *Transactions on Emerging Telecommunications Technologies*, 31(November 2019):1–24, 2020.

[31] M. Goyal, E. Baccelli, M. Philipp, A. Brandt, and J. Martocci. Reactive discovery of point-to-point routes in low power and lossy networks. *IETF Request For Comments RFC*, 6997, 2013.

[32] M. Ha, S. H. Kim, and D. Kim. Intra-mario: A fast mobility management protocol for 6lowpan. *IEEE Transactions on Mobile Computing*, 16(1):172–184, 2016.

[33] J. D. Hol, T. B. Schon, and F. Gustafsson. On resampling algorithms for particle filters. In *2006 IEEE nonlinear statistical signal processing workshop*, pages 79–82. IEEE, 2006.

[34] F. F. Jurado Lasso, L. Marchegiani, J. F. Jurado, A. A. Mahfouz, and X. Fafoutis. A survey on software-defined wireless sensor networks: Current status, machine learning approaches and major challenges. *PrePrint*, 2021.

[35] P. O. Kamgueu, E. Nataf, and T. D. Ndie. Survey on RPL enhancements: A focus on topology, security and mobility. *Computer Communications*, 120:10–21, 2018.

[36] P. O. Kamgueu, E. Nataf, T. D. Ndie, P. O. Kamgueu, E. Nataf, T. Djotio, and N. Survey. Survey on RPL enhancements : a focus on topology , security and mobility To cite this version : HAL Id : hal-01713247. *Computer Communications*, 2018.

[37] H. Kharrufa, H. Al-Kashoash, and A. H. Kemp. A game theoretic optimization of rpl for mobile internet of things applications. *IEEE Sensors Journal*, 18(6):2520–2530, 2018.

[38] J. Ko and M. Chang. Momoro: Providing mobility support for low-power wireless applications. *IEEE Systems Journal*, 9(2):585–594, 2014.

[39] H. I. Kobo, A. M. Abu-Mahfouz, and G. P. Hancke. A survey on software-defined wireless sensor networks: Challenges and design requirements. *IEEE access*, 5:1872–1899, 2017.

[40] R. Labbe. Kalman and bayesian filters in python, 2014. *URL https://github. com/rlabbe/Kalman-and-Bayesian-Filters-in-Python*, 2019.

[41] Z. Latif, K. Sharif, F. Li, M. M. Karim, S. Biswas, and Y. Wang. A comprehensive survey of interface protocols for software defined networks. *Journal of Network and Computer Applications*, 156:102563, 2020.

[42] K. C. Lee, R. Sudhaakar, J. Ning, L. Dai, S. Addepalli, J. Vasseur, and M. Gerla. A comprehensive evaluation of RPL under mobility. *International Journal of vehicular technology*, 2012, 2012.

[43] P. Levis, T. Clausen, J. Hui, O. Gnawali, and J. Ko. The trickle algorithm. *Internet Engineering Task Force, RFC6206*, 2011.

[44] P. Levis, S. Madden, J. Polastre, R. Szewczyk, K. Whitehouse, A. Woo, D. Gay, J. Hill, M. Welsh, E. Brewer, et al. Tinyos: An operating system for sensor networks. In *Ambient intelligence*, pages 115–148. Springer, 2005.

[45] L. Li, W. Yang, and G. Wang. Intelligent fusion of information derived from received signal strength and inertial measurements for indoor wireless localization. *AEU - International Journal of Electronics and Communications*, 70(9):1105–1113, 2015.

[46] T. Luo, H.-P. Tan, and T. Q. Quek. Sensor openflow: Enabling software-defined wireless sensor networks. *IEEE Communications letters*, 16(11):1896–1899, 2012.

[47] J. Mertens, G. Milotta, P. Nagaradjane, and G. Morabito. Sdn-(uav) ise: Applying software defined networking to wireless sensor networks with data mules. In *2020 IEEE 21st International Symposium on" A World of Wireless, Mobile and Multimedia Networks"(WoWMoM)*, pages 323–328. IEEE, 2020.

[48] L. Mihaylova, D. Angelova, S. Honary, D. R. Bull, C. N. Canagarajah, and B. Ristic. Mobility tracking in cellular networks using particle filtering. *IEEE Transactions on Wireless Communications*, 6(10):3589–3599, 2007.

[49] A. Mohammadsalehi, B. Safaei, A. M. H. Monazzah, L. Bauer, J. Henkel, and A. Ejlali. ARMOR: A Reliable and Mobility-aware RPL for Mobile Internet of Things Infrastructures. *IEEE Internet of Things Journal*, pages 1–1, jun 2021.

[50] R. Morabito and J. Jiménez. Ietf protocol suite for the internet of things: Overview and recent advancements. *IEEE Communications Standards Magazine*, 4(2):41–49, 2020.

[51] S. Murali and A. Jamalipour. Mobility-aware energy-efficient parent selection algorithm for low power and lossy networks. *IEEE Internet of Things Journal*, 6(2):2593–2601, 2018.

[52] M. J. Neely. Stochastic network optimization with application to communication and queueing systems. *Synthesis Lectures on Communication Networks*, 3(1):1–211, 2010.

[53] A. Oliveira and T. Vazao. Low-power and lossy networks under mobility: A survey. *Computer networks*, 107:339–352, 2016.

[54] C. Orfanidis, P. Pop, and X. Fafoutis. Active connectivity fundamentals for tsch networks of mobile robots. *arXiv preprint arXiv:2201.08202*, 2022.

[55] F. Orozco-Santos, V. Sempere-Payá, J. Silvestre-Blanes, and T. Albero-Albero. Multicast scheduling in sdn wise to support mobile nodes in industrial wireless sensor networks. *IEEE Access*, 9:141651–141666, 2021.

[56] F. Österlind. A sensor network simulator for the contiki os. *SICS Research Report*, 2006.

[57] F. Osterlind, A. Dunkels, J. Eriksson, N. Finne, and T. Voigt. Cross-level sensor network simulation with cooja. In *Proceedings. 2006 31st IEEE conference on local computer networks*, pages 641–648. IEEE, 2006.

[58] A. Panisson. pymobility v0.1 - python implementation of mobility models, 2014.

[59] R. Parasuraman, P. Oegren, and B. C. Min. Kalman Filter Based Spatial Prediction of Wireless Connectivity for Autonomous Robots and Connected Vehicles. *IEEE Vehicular Technology Conference*, 2018-Augus, 2018.

[60] J. Park, K. H. Kim, and K. Kim. An algorithm for timely transmission of solicitation messages in RPL for energy-efficient node mobility. *Sensors (Switzerland)*, 17:1–21, 2017.

[61] A. K. Paul and T. Sato. Localization in wireless sensor networks: A survey on algorithms, measurement techniques, applications and challenges. *Journal of sensor and actuator networks*, 6(4):24, 2017.

[62] I. Rabet, H. Fotouhi, M. Vahabi, M. Björkman, and M. Alves. Rpl-rp: Rpl with route projection for transversal routing. In *IEEE World Forum IoT 2021*, 2021.

[63] I. Rabet, S. P. Selvaraju, M. H. Adeli, H. Fotouhi, A. Balador, M. Vahabi, M. Alves, and M. Björkman. Pushing iot mobility management to the edge: Granting rpl accurate localization and routing. In *2021 IEEE 7th World Forum on Internet of Things (WF-IoT)*, pages 338–343. IEEE, 2021.

[64] I. Rabet, S. P. Selvaraju, H. Fotouhi, M. Vahabi, and M. Bjorkman. Poster: Particle Filter for Handoff Prediction in SDN-based IoT Networks. In *EWSN 2020*, 2019.

[65] I. Rabet, S. P. Selvaraju, H. Fotouhi, M. Vahabi, and M. Björkman. Poster: Particle filter for handoff prediction in sdn-based iot networks. In *EWSN*, pages 172–173, 2020.

[66] T. Rappaport. Wireless communications: principles and practice, Vol. 2. New Jersey: Prentice Hall PTR,, 1996.

[67] I. Rhee, M. Shin, S. Hong, K. Lee, S. J. Kim, and S. Chong. On the levy-walk nature of human mobility. *IEEE/ACM transactions on networking*, 19(3):630–643, 2011.

[68] E. M. Royer, P. M. Melliar-Smith, and L. E. Moser. An analysis of the optimum node density for ad hoc mobile networks. In *ICC 2001. IEEE International Conference on Communications. Conference Record (Cat. No. 01CH37240)*, volume 3, pages 857–861. IEEE, 2001.

[69] A. R. J. Ruiz and F. S. Granja. Comparing ubisense, bespoon, and decawave uwb location systems: Indoor performance analysis. *IEEE Transactions on instrumentation and Measurement*, 66(8):2106–2117, 2017.

[70] B. Safaei, A. Mohammadsalehi, K. T. Khoosani, S. Zarbaf, A. M. H. Monazzah, F. Samie, L. Bauer, J. Henkel, and A. Ejlali. Impacts of mobility models on rpl-based mobile iot infrastructures: An evaluative comparison and survey. *IEEE Access*, 8:167779–167829, 2020.

[71] O. Safety, D. Health Administration, Washington, U. S. O. Safety, H. Administration, L. D. O. Safety, H. Administration, et al. *Training requirements in OSHA standards and training guidelines*. Number 2254. Occupational Safety & Health Administration, 1998.

[72] S. Särkkä. *Bayesian filtering and smoothing*, volume 3. Cambridge University Press, 2013.

[73] B. Shao and M. S. Leeson. PaFiR: Particle Filter Routing–A predictive relaying scheme for UAV-assisted IoT communications in future innovated networks. *Internet of Things*, page 100077, 2019.

[74] Z. Sheng, S. Yang, Y. Yu, A. V. Vasilakos, J. A. McCann, and K. K. Leung. A survey on the ietf protocol suite for the internet of things: Standards, challenges, and opportunities. *IEEE wireless communications*, 20(6):91–98, 2013.

[75] Y. Tahir, S. Yang, and J. McCann. Brpl: Backpressure rpl for high-throughput and mobile iots. *IEEE Transactions on Mobile Computing*, 17(1):29–43, 2017.

[76] O. Tavallaie, J. Taheri, and A. Y. Zomaya. Design and optimization of traffic-aware tsch scheduling for mobile 6tisch networks. In *Proceedings of the International Conference on Internet-of-Things Design and Implementation*, pages 234–246, 2021.

[77] T. Theodorou and L. Mamatas. Sd-miot: A software-defined networking solution for mobile internet of things. *IEEE Internet of Things Journal*, 8(6):4604–4617, 2020.

[78] P. Thubert. Objective function zero for the routing protocol for low-power and lossy networks (rpl [online] available: https://tools. ietf. org/html/rfc6552. *Accessed: April*, 2017.

[79] P. Thubert, R. Jadhav, and M. Gillmore. Root initiated routing state in RPL. RFC-draft 1654, RFC Editor, July 2020.

[80] P. Thubert, R. Jadhav, and G. M. Root initiated routing state in RPL draft-ietf-roll-dao-projection-09, 2019.

[81] P. Thubert, M. R. Palattella, and T. Engel. 6TiSCH centralized scheduling: When SDN meet IoT. In *2015 IEEE conference on standards for communications and networking (CSCN)*, pages 42–47. IEEE, 2015.

[82] P. Thubert and M. Richardson. Routing for RPL leaves. *Work in Progress, draft-thubert-roll-unaware-leaves-05*, 2018.

[83] B. Tian, K. M. Hou, H. Shi, X. Liu, X. Diao, J. Li, Y. Chen, and J.-P. Chanet. Application of modified RPL under VANET-WSN communication architecture. In *2013 international conference on computational and information sciences*, pages 1467–1470. IEEE, 2013.

[84] G. Violettas, S. Petridou, and L. Mamatas. Routing under Heterogeneity and Mobility for the Internet of Things: A Centralized Control Approach. In *2018 IEEE Global Communications Conference, GLOBECOM 2018 - Proceedings*, 2018.

[85] T. Winter, P. Thubert, A. Brandt, J. W. Hui, R. Kelsey, P. Levis, K. Pister, R. Struik, J.-P. Vasseur, R. K. Alexander, et al. RPL: IPv6 Routing Protocol for Low-Power and Lossy Networks. *rfc*, 6550:1–157, 2012.

[86] I. Yaqoob, E. Ahmed, I. A. T. Hashem, A. I. A. Ahmed, A. Gani, M. Imran, and M. Guizani. Internet of things architecture: Recent advances, taxonomy, requirements, and open challenges. *IEEE wireless communications*, 24(3):10–16, 2017.

[87] C. Zhang, P. Patras, and H. Haddadi. Deep Learning in Mobile and Wireless Networking: A Survey. *IEEE Communications Surveys and Tutorials*, 21(3):2224–2287, 2019.

[88] M. Zuniga and B. Krishnamachari. Analyzing the transitional region in low power wireless links. In *2004 First Annual IEEE Communications Society Conference on Sensor and Ad Hoc Communications and Networks, 2004. IEEE SECON 2004.*, pages 517–526. IEEE, 2004.

Part II

Included Papers

Chapter 5

Paper A: MobiFog: Mobility Management Framework for Fog-assisted IoT Networks

Hossein Fotouhi, Maryam Vahabi, Iliar Rabet, Mats Björkman, and Mário Alves In the proceedings of IEEE Global Conference on Internet of Things (GCIoT) 2019.

Abstract

Mobility is becoming a challenging issue in upcoming IoT applications, where it is crucial to employ mobile entities. In parallel, Fog computing has revolutionized network architecture, while enabling local processing of measurements, and reducing bandwidth overhead, which results in a more reliable system and real-time support. However, mobility management is a missing framework within the mobile IoT networks with Fog computing architecture. This paper provides a simple and generic seamless handoff model, dubbed as MobiFog, where it addresses the handoff mechanism with zero delay, while providing high reliability.

5.1 Introduction

The current trends in the Internet of Things (IoT) show a growing need for flexibility. Many emerging and future applications will require or at least benefit from more heterogeneous devices (different hardware/software platforms) that may (or must) physically move. This means that IoT networks will experience more topological changes [1]. This can result in the loss of important sensing data or imposing large delays for time-sensitive applications such as healthcare monitoring and factory automation.

A typical mobility solution is to implement a handoff mechanism, where mobile nodes are supposed to switch from one point of attachment to another; these points of attachment are usually known as Access Points (APs) and the network of APs is so called fixed infrastructure. Conventional handoff mechanisms have been designed mostly in a distributed manner, where mobile nodes are assumed to perform 100% of handoff duties [2–4]. Using an SDN controller for mobility management will require the redesign of the handoff mechanism.

Besides the resource constraint property of IoT networks, lack of centralized network management would increase the complexity, specially when it comes to networks with mobile nodes. Software-defined networking (SDN) centralizes network control, and provides dynamicity, flexibility and reconfigurability within the network. The main concept of SDN relies on the notion of separating the control and data plane.

There have been many efforts to design SDN controllers for IoT networks, specifically for sensor networks [5–7]. However, all these works were quite preliminary, lacking real algorithm design and evaluation in COTS/standard platforms. We have previously designed and evaluated a simplistic design of SDN for IoT networks [8]. This work has been conducted for a small-scale network in order to avoid network congestion by providing a traffic-aware solution applied to the SDN controller [9].

Contribution. This paper presents a mobility management framework for a Fog-based IoT network, focusing on a novel design of a seamless handoff approach that provides zero handoff delay with high reliability.

Paper organization. Section 5.2 presents the related works on mobility management. Section 5.3 gives a general model of the proposed approach (MobiFog) to support seamless handoff mechanism and its analytical evaluation and section 5.4 concludes the paper.

5.2 Literature review

There have been many efforts on devising mobility management solutions for IoT and wireless sensor networks. In this section we review state of the art, identify existing gaps and key research questions to be answered when designing a Fog-based mobility mechanism.

Previously, we have proposed hard and soft handoff approaches for conventional IoT networks, where these ideas are partially used in the MobiFog model.

Smart-HOP. This algorithm has two main phases of Data Transmission and Discovery. The Mobile Node (MN) periodically monitors the link quality level by receiving beacons from the serving AP (current parent node). Upon receiving a number of data packets in a given window, the serving AP replies with the average received signal strength (ARSSI) or average signal to noise ratio (ASNR).

The MN disconnects from the serving AP when the link quality degrades or breaks (no packet reception). Thus, MN immediately enters the Discovery Phase with broadcasting burst of beacons to neighbor APs, while expecting replies after each burst. When a high link quality (ARSSI greater than a threshold level) is detected, the MN attaches to the new serving AP.

mRPL. mRPL is the smart-HOP algorithm integrated within the standard RPL routing. The major changes are: (i) the use of RPL control messages instead of beacons (i.e. DIS and DIO messages), and (ii) the additional timers to increase handoff efficiency and reliability. The process of dropping, keeping and assessing link(s) are similar to the smart-HOP mechanism.

Unlike smart-HOP, the MN keeps data communication with the serving AP until it finds a better AP. After a successful handoff, the nullifying process of the RPL algorithm is executed.

mRPL+. mRPL+ is a combination of the hard and soft handoff approaches within the RPL routing. The main contribution of mPL+ the introduction of overhearing mechanism. The APs overhear the link (between MN and the serving AP) activities by measuring the average RSSI of the packets received from the MN. The chipcon CC2420 MAC sub-layer supports promiscuous mode that sniffs packets.

Handoff in conventional wireless architecture is classified into two main categories: hard and soft handoffs. In a hard handoff, MN disconnects from one AP and searches for a new AP, which implies a disconnection period. Consequently, a hard handoff mechanism is prone to high packet losses. In a soft handoff, the new potential AP is selected while communicating with the current AP. [4]

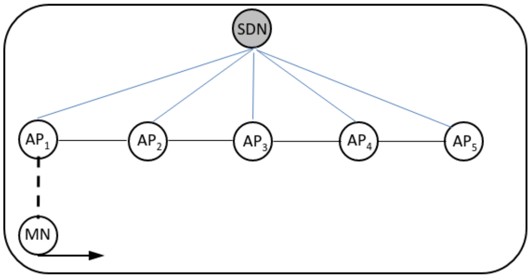


Figure 5.1: A Fog computing IoT architecture with an SDN controller, a set of APs and a mobile node.

Several other works focused on mobility support in commodity IoT protocols by enhancing 6loWPAN, RPL, and CoAP [10–13]. A recent survey paper collects some of the related works in this area [14]. The main issue of these works is the special focus on one of the network layers, and the lack of ability to re-adapt to architectural changes.

5.3 Mobility management framework

In this work, we outline the design of a mobility management mechanism to support seamless zero-delay handoffs. It also ensures that the data packets reach to the destination through multiple paths.

MobiFog handoff mechanism. Unlike our previous models, where the system was divided into two phases of Discovery and Data Transmission, these phases are merged together. Thus, discovery of alternative APs and data transmission are performed in parallel. The main difference in the current model is the existence of superframe notion that implies periodic *beaconing*, initiated by the SDN controller. The beaconing phase gives the opportunity of distributing some information regarding MN's neighbors. Figure 5.1 gives an example of an SDN-based IoT network with five APs and a MN, where the MN travels from the vicinity of AP_1 toward AP_5, while passing AP_2, AP_3 and AP_4. Figure 5.2 shows the transitions and the packets exchanged while MN moves.

The MN receives its neighbor list from the SDN controller during the beaconing phase. This list enforces MN to multicast its data to all the APs in the list, while keeping one of them as its proffered parent node. This way, the MN ensures that the data has been successfully transmitted to the fixed infrastructure.

Assuming that initially MN is connected to AP_1, it sends its data to the

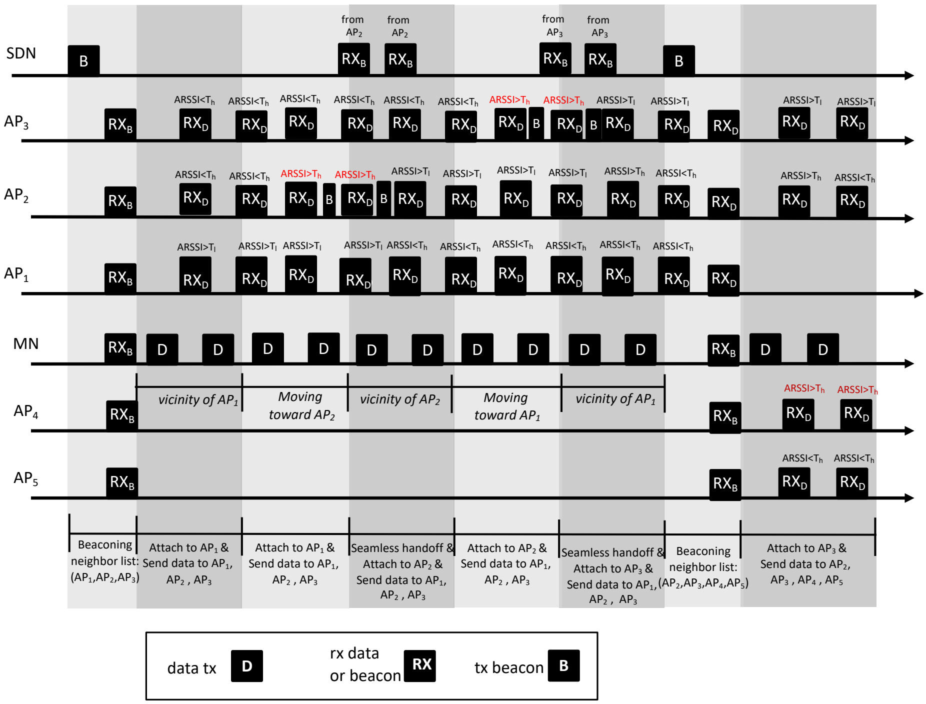


Figure 5.2: As the MN moves across the horizontal path, the corresponding ARSSI changes over time and there is no need for any beaconing in the seamless handoff but if the parent that is the MN is going to attach to, does not exist in the MN's parent list a beacon from the SDN is required.

parent list [AP_1,AP_2,AP_3]. By moving towards AP_2, the AP with the highest link quality (AP_2 in this example), sends a beacon to SDN controller indicating its ability for future service to the MN. SDN controller sends a new policy to the MN and the neighbor list in order to execute the handoff mechanism. Then MN immediately switches from AP_1 to AP_2, while sending data to the same list of neighbors. This mechanism repeats until MN requires a new set of neighbor APs. There is a need for fine tuning the beaconing period as it is responsible for updating the AP list, and consequently, it affects the mobility management framework. After receiving new commands through beacons from the SDN controller, the neighbor list will be updated. In this example, the new parent list comprises AP_2, AP_3, AP_4 and AP_5.

MobiFog analytical model. Probabilistic modelling provides a more structured view of the handoff algorithm and facilitates performance analysis. By holding environmental parameters, it provides the opportunity for further analysis of the model before testing/validating the algorithm through simulation and experimental models. Two main channel parameters are known as: (i) path-loss exponent (η) that measures the power of radio frequency signals relative to distance, and (ii) standard deviation (σ) that measures the standard deviation in RSSI measurements due to log-normal shadowing. The values of η and σ change with the frequency of operation and the clutter and disturbance in the environment. We assume a scenario similar to the network described in figure 5.1. Considering the handoff between two APs (AP_a and AP_b) could be any of the APs in the scenario.

In this work, we formulate the probability of starting a seamless handoff mechanism. For the sake of simplicity, we assume that RSSI is the link quality metric. The probabilities of being below the lower threshold level (T_ℓ) and above the higher threshold (T_h) level are defined by using a Q-function. The traveling path of the MN is divided into a number of slots. These probabilities are expressed as follows.

$$P(R_a(i) < T_\ell) = Q(\frac{-T_\ell + R_a(i)}{\sigma})$$

$$P(R_b(i) > T_h) = Q(\frac{T_h - R_b(i)}{\sigma})$$

Where $Q(.)$ is the complementary distribution function of the standard Gaussian, i.e., $Q(x) = \int_x^\infty (1/\sqrt{2\pi}) e^{-t^2/2} dt$, $R_a(i)$ and $R_b(i)$ indicates the RSSI values from AP_a and AP_b at slot i, and σ (in dB) expresses the standard deviation.

The received signal strength is estimated by a log-normal shadowing path-loss. According to this model, $R(i)$ (in dBm) (RSSI level at a given slot i) from the transmitter is given by [15]:

$$R(i) = P_t - \overline{PL(d_0)} - 10\eta log_{10}(i/d_0) - X_\sigma \tag{5.1}$$

Where i corresponds to distance, P_t is the transmission power, $\overline{PL(d_0)}$ is the measured path-loss at reference distance d_0, n is the path-loss exponent, and $X_\sigma = N(0, \sigma)$ is a Normal variable (in dB). The term X_σ models the path-loss variation across all locations at distance i from the source due to shadowing, a term that encompasses signal strength variations due to the characteristics of the environment (i.e., occlusions, reflections, etc.).

The probability of starting a hard handoff at slot $s \in [1, k)$ is defined as follows (k indicates the total number of slots).

$$P_{\text{seamless}}(S(s)) = \left[\prod_{i=s-2w-2}^{s-w-2} (P(R_a(i) \geq T_\ell) \times P(R_b(i) < T_h)) \right] \times \left[\prod_{i=s-w-1}^{s-1} (P(R_a(i) \geq T_\ell) \times P(R_b(i) \geq T_h)) \right] \tag{5.2}$$

The first part of the equation indicates that the MN is connected to AP_a, sending data to both AP_a and AP_b, while observing low link quality with AP_b. This observation is performed for a time span of window size w. The second part of the equation reflects the situation where the MN is still connected to AP_a, and experiencing a high link quality with the neighbor AP, while still sending data to both APs. By experiencing such situation in two consecutive window sizes, MN will start the seamless handoff mechanism at slot s.

Equation 5.3 formulates the probability of ending the seamless handoff at slot $e \in (s, k]$, considering the fact that the handoff mechanism has started at slot s. Note that in this mechanism, MN starts connecting to new AP before disconnecting from the current AP. $P_{\text{Seamless}}(E(e) \mid S(s)) =$

$$\begin{cases} P(R_b(e) \geq T_h) & e = s \\ \left[\prod_{i=s}^{e-1} P(R_b(i) \leq T_h)\right] \\ \times P(R_b(e) \geq T_h), & e \geq s \end{cases} \tag{5.3}$$

As the first part of the equation indicates, there is a higher probability to experience a zero delay in the seamless handoff, since the only necessary condition to end the handoff is that the RSSI of AP_b is higher than T_h. In the second part of the equation, if AP_b does not have a good link after starting the seamless mechanism, then the seamless handoff ends as soon as there is a link with a RSSI above T_h.

Based on Equation 5.4, the handoff delay can reach to zero when starting and ending slots are happening at the same moment. The expected handoff delay is computed by getting the weighted sum of all possible handoff periods. It is defined as the product of the time spent in each possible handoff mechanism started at slot s and ended at slot e by the correspondent probabilities of starting a handoff at slot s, $P(S(s))$, and ending it at slot e, $P(E(e) \mid S(s))$. For each handoff starting at slot s, the handoff would end at one of the slots from $s + 1$ to k. The sum of all these possible situations defines the expected delay for a handoff started at a specific slot s. The overall expected handoff delay is defined as follows. $\text{Delay}(s, e) =$

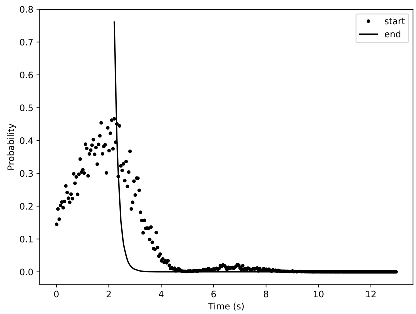


Figure 5.3: Probability of starting handoff, gets to its maximum after 2.2 seconds. Assuming that handoff starts at 2.2, the conditional probability of ending the handoff, is maximized at exactly the same time instance.

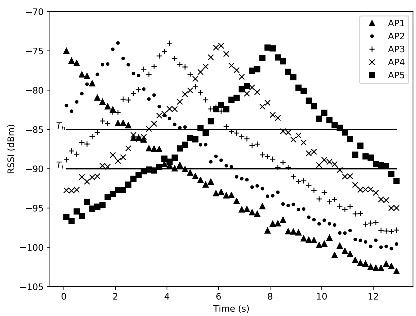


Figure 5.4: The RSSI measured at the APs when the MN moves with a constant speed of 1m/s and the APs are statically placed, as in Figure 5.1

$$\sum_{s=1}^{k-1} \sum_{e=s+1}^{k} ((e-s) \times P(E(e) \mid S(s)) \times P(S(s))) \tag{5.4}$$

The following settings are used in the analytical evaluations across this section: $\theta = 4$dB, $\eta = 4$, $P_t = 0$dBm, $d_0 = 1$m, $d = 6$m, $P_l(d_0) = -55$dB, $T_l = -90$dBm and $T_h = -85$dBm.

Most often, there is more than one reachable AP to be indexed in the parent list by the SDN controller and facilitate the handoff mechanism. If the parent that MN is going to attach to, does not exist in the parent list, it will be required that the controller sends a beacon to add it to MN's parent list, otherwise a zero delay is expected. But in some cases, as in 13th second, the RSSI value of all of the APs drops below the T_ℓ, then the MN needs to call for a hard handoff which would imply a higher delay roughly 100ms.

Considering a handoff from AP_1 to AP_3, one can calculate the probability of starting the seamless handoff and the conditional probability of ending the handoff as shown in Figure 5.3. The probability of starting handoff gets to it maximum around time 2.2 second, therefore, the conditional probability of ending the handoff is at its maximum at the same time slot. Overall, the handoff delay is reduced to zero in seamless handoff mechanism compared to average 4ms in soft handoff in mRPL+ and 10ms in hard handoff.

5.4 Concluding remarks

This paper highlights the need for mobility support in future Internet-of-Things (IoT) applications, outlining an innovative architecture - dubbed as MobiFog. MobiFog builds on separated data and control planes, where the latter is managed by an SDN controller. We summarise our previous handoff approaches (smartHOP, mRPL and mRPL+) and elaborate on the MobiFog seamless handoff mechanism. Overhearing in the APs enables them to notify the SDN controller about the status of their links to mobile nodes, enabling the SDN controller to proactively update the RPL parent list in the mobile nodes. We also provide a probabilistic analysis of the probability of starting and ending a handoff, and as the results of the analysis shows, there is a very high probability for the delay to be zero so the expected delay would be dramatically lower compared to the 4ms in soft handoff in mRPL+. A valuable future work would be to study the challenges to devise an optimized parent list.

Bibliography

[1] Ibrar Yaqoob, Ejaz Ahmed, Ibrahim Abaker Targio Hashem, Abdelmuttlib Ibrahim Abdalla Ahmed, Abdullah Gani, Muhammad Imran, and Mohsen Guizani. Internet of things architecture: Recent advances, taxonomy, requirements, and open challenges. *IEEE wireless communications*, 24(3):10–16, 2017.

[2] Hossein Fotouhi, Marco Zuniga, Mário Alves, Anis Koubâa, and Pedro Marrón. Smart-hop: A reliable handoff mechanism for mobile wireless sensor networks. In *European Conference on Wireless Sensor Networks*, pages 131–146. Springer, 2012.

[3] Hossein Fotouhi, Daniel Moreira, and Mário Alves. mRPL: Boosting mobility in the Internet of Things. *Ad Hoc Networks*, 26:17–35, 2015.

[4] Hossein Fotouhi, Daniel Moreira, Mário Alves, and Patrick Yomsi. mRPL+: A mobility management framework in RPL/6LoWPAN. *Computer Communications*, 104:34–54, 2017.

[5] Tie Luo, Hwee-Pink Tan, and Tony QS Quek. Sensor openflow: Enabling software-defined wireless sensor networks. *IEEE Communications letters*, 16(11):1896–1899, 2012.

[6] Alejandro De Gante, Mohamed Aslan, and Ashraf Matrawy. Smart wireless sensor network management based on software-defined networking. In *2014 27th Biennial Symposium on Communications (QBSC)*, pages 71–75. IEEE, 2014.

[7] Laura Galluccio, Sebastiano Milardo, Giacomo Morabito, and Sergio Palazzo. SDN-WISE: Design, prototyping and experimentation of a stateful sdn solution for WIreless SEnsor networks. In *2015 IEEE Conference on Computer Communications (INFOCOM)*, pages 513–521. IEEE, 2015.

[8] Hossein Fotouhi, Maryam Vahabi, Apala Ray, and Mats Björkman. SDN-TAP: an SDN-based traffic aware protocol for wireless sensor networks. In *2016 IEEE 18th International Conference on e-Health Networking, Applications and Services (Healthcom)*, pages 1–6. IEEE, 2016.

[9] Adam Dunkels, Oliver Schmidt, Niclas Finne, Joakim Eriksson, F Österlind, Nicolas Tsiftes, and Mathilde Durvy. The Contiki OS: The operating system for the internet of things, 2011, 2018.

[10] Minkeun Ha, Seong Hoon Kim, and Daeyoung Kim. Intra-mario: A fast mobility management protocol for 6lowpan. *IEEE Transactions on Mobile Computing*, 16(1):172–184, 2016.

[11] Gargi Bag, Muhammad Raza, Ki-Hyung Kim, and Seung-Wha Yoo. Lowmob: Intra-pan mobility support schemes for 6lowpan. *Sensors*, 9(7):5844–5877, 2009.

[12] JeongGil Ko and Marcus Chang. Momoro: Providing mobility support for low-power wireless applications. *IEEE Systems Journal*, 9(2):585–594, 2014.

[13] Olfa Gaddour, Anis Koubäa, Raghuraman Rangarajan, Omar Cheikhrouhou, Eduardo Tovar, and Mohamed Abid. Co-rpl: Rpl routing for mobile low power wireless sensor networks using corona

mechanism. In *Proceedings of the 9th IEEE international symposium on industrial embedded systems (SIES 2014)*, pages 200–209. IEEE, 2014.

[14] Patrick Olivier Kamgueu, Emmanuel Nataf, and Thomas Djotio Ndie. Survey on RPL enhancements: A focus on topology, security and mobility. *Computer Communications*, 120:10–21, 2018.

[15] T.S. Rappaport. Wireless communications: principles and practice, Vol. 2. New Jersey: Prentice Hall PTR,, 1996.

Chapter 6

Paper B: Poster: Particle Filter for Handoff Prediction in SDN-based IoT Networks

Iliar Rabet, Shunmuga Priyan Selvaraju, Hossein Fotouhi, Maryam Vahabi and Mats Björkman
In the International Conference on Embedded Wireless Systems and Networks (EWSN)

Abstract
Standard implementation of RPL protocol has struggled to limit the impact of mobility on the throughput of the IoT network. Handoff process is of great importance to optimize the trade-off between the control overhead (for maintaining the network topology), and the delay, caused by nodes mobility. In this work, We have proposed a method for predicting future handoffs through fusion of RSSI value and Inertial Measurement Unit (IMU) information using particle filter, which is known for accuracy albeit it needs higher computation capacity. The provided analytical model indicates lower network interruption with the proposed method.

6.1 Introduction

Mobility is one of the major elements in future Internet of Things (IoT) applications, which requires proper design of mobility solutions for IoT networks and protocols. Most of the IoT standards such as Routing Protocol for Low-Power and Lossy Networks (RPL) and 6LoWPAN assume that topological changes are negligible. Experiments have revealed that RPL experiences a degraded throughput in a mobile environment [1–3].

Smart-hop [1], mRPL [2] and mRPL+ [3] proposed handoff processes with low communication overhead and low handoff delay for IoT networks. These handoff processes were designed in such a way to be independent of the mobile node movement information. This issue may cause handoff performance degradation in networks, where the mobile node changes its mobility pattern. In some cases, it may also reduce network responsiveness to dynamic changes. Barcleo et. al. [4] have proposed Kalman Position-RPL (KP-RPL) in which, each mobile node is equipped with IMU sensors for sending positioning beacons regardless to velocity with a constant interval, and then they predict the future position based on Kalman filter. Similarly, EKF-MRPL [5] uses extended Kalman filter to predict non-linear trajectory paths. In [6] authors utilize Doppler effect to estimate the velocity of the mobile nodes. This approach may fail to detect mobility in case there is no radio activity.

Particle filter, also known as sequential Monte Carlo [7], is a method used for non-linear filtering problems. Kalman filter is unable to provide good performance in scenarios with non-linear relation and non-Gaussian noise, and its variations can handle such scenarios with low accuracy. In this work, we have designed a particle filter with higher accuracy due to the number of samples, that predicts the future Received Signal Strength Indicator (RSSI). Regarding the hardware constraints in IoT devices, most standards have been designed to avoid computation-heavy tasks in distributed nodes. With the rise of Fog computing, it is possible to offload the computation, as in MobiFog [8], which is a novel Fog-based approach designed for RPL routing. In this approach, a centralised SDN controller manages the parent list of all nodes in the network, and provides a seamless handoff process with zero delay.

6.2 Prediction Framework

In many deployments, IMU sensors are used in application layer, and it is possible to make use of them in routing layer, or provide a cross layer approach. In our proposed method, when mobile nodes start moving, they broadcast their velocity to all static nodes in their vicinity in a specific interval. Mobile node

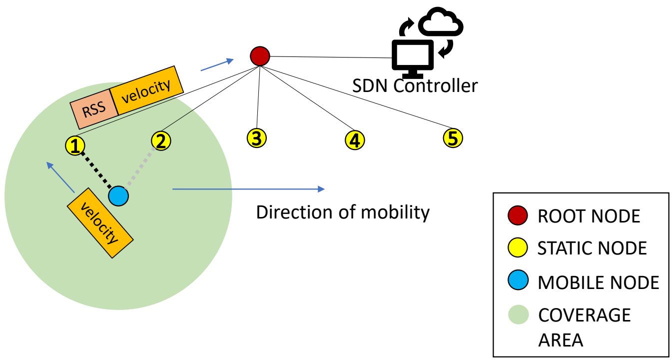


Figure 6.1: As the mobile node moves the SDN controller updates the routes in the static and mobile node.

stops sending broadcasts as soon as it stops moving. However, in standard RPL, a mechanism called Trickle is used to determine the interval between the control messages to capture any change in the topology. Trickle algorithm gradually restores its interval after each change in the topology, but in our approach the interval restoration can be done instantaneously.

As shown in Figure 6.1, the intermediary nodes that can receive the broadcast packet including the velocity information, append the RSSI to the packet and forward it to the controller. The controller predicts the RSSI values in the next time slot and based on that, the routing manager decides to update the parent list using a SDN control packet.

As shown in Figure 6.2, we maintain the state information including both position and RSSI values from all static nodes that have received the broadcast message. The prediction step in the filter can use equation 6.1 to estimate future RSSI.

$$RSSI_D - RSSI_{D0} = -10\alpha.log(\frac{D0}{D}) \tag{6.1}$$

6.3 Analytical Model

We devised an analytical model to compare performance of different behaviors of the system. There are also some research works that model the consistency of the standards [9]. However our model aims at comparing packet loss in a mobile scenario between standard RPL and the proposed method. The scenario consists of one mobile node that moves from vicinity of static node A and

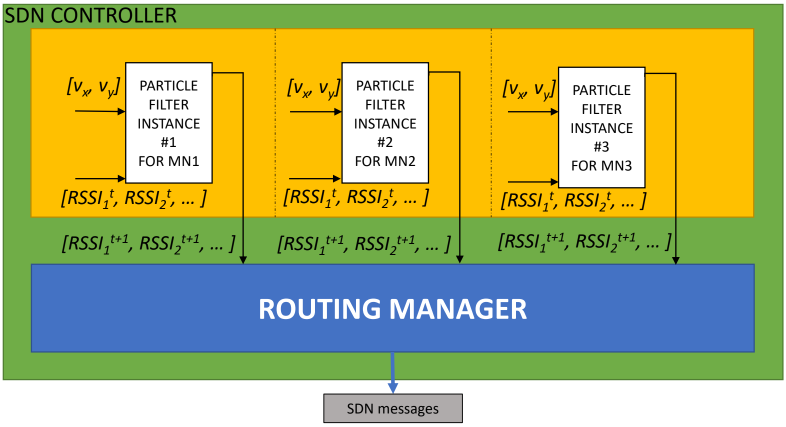


Figure 6.2: The particle filter can have the current RSSI values and velocity of the mobile node as input and predicts the future RSSI values at different static nodes. For each mobile node there can one specific instance of filter.

decreases the distance to another static node B. We assume that the mobile node is moving with a constant speed of 1 m/s horizontally and travels for 12 seconds and $T_\ell = -90$ dBm. Thus the probability of the RSSI to be below the threshold for starting a handoff process is defined as:

$$P(R_a(i) < T_\ell) = Q(\frac{-T_\ell + R_a(i)}{\sigma})$$

Q function is the complementary distribution function of the standard Gaussian. The probability of packet loss at time t is proportional to the probability of the RSSI being below T_ℓ, while knowing that handoff is not initiated yet.

$$P_{Loss}(t) \propto P(R_a(i) < T_\ell) \times (1 - \int_0^t P_{handoff}(\theta)d\theta)$$

For the probability of starting handoff in standard RPL, if the mobile node starts moving when the network has been static for a while, then the Trickle interval would have a higher value, e.g. 200 sec. In this case, Trickle algorithm decides a value between 100 and 200 with a uniform distribution. So it is unlikely that any handoff occurs, and hence there is high probability of packet loss. But if the timer was set to a lower value, e.g. 2 sec, then the probability of handoff would be a uniform distribution between 1 to 2 sec after detecting the link breakdown. In our proposed method, the handoff predicts the link quality in a proactive manner, so it will start the process without any delay and

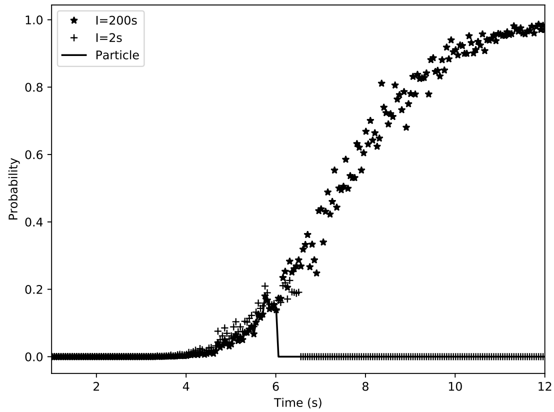


Figure 6.3: Probability of packet loss when mobile node starts moving, with trickle interval at 200 sec (so long that prohibits fast detection of topology changes). With a 2 sec Trickle interval, the topology change is detected but almost 1 sec after the particle filter prediction.

experiences a lower packet loss as shown in Figure 6.3.

6.4 Conclusions

We have proposed a handoff prediction mechanism using particle filter to facilitate seamless handoff. The computation is offloaded to a centralised Fog node, which has a global knowledge of the networks. Using IMU sensors, we can also optimize the Trickle interval, which has a long lasting effect on the power consumption in the network. As indicated by the analytical model, the proposed mechanism leads to shorter network interruption during the handoff.

Bibliography

[1] Hossein Fotouhi, Marco Zuniga, Mário Alves, Anis Koubâa, and Pedro Marrón. Smart-hop: A reliable handoff mechanism for mobile wireless sensor networks. In *European Conference on Wireless Sensor Networks*, pages 131–146. Springer, 2012.

[2] Hossein Fotouhi, Daniel Moreira, and Mário Alves. mRPL: Boosting mobility in the Internet of Things. *Ad Hoc Networks*, 26:17–35, 2015.

[3] Hossein Fotouhi, Daniel Moreira, Mário Alves, and Patrick Yomsi. mRPL+: A mobility management framework in RPL/6LoWPAN. *Computer Communications*, 104:34–54, 2017.

[4] Marc Barcelo, Alejandro Correa, Jose Lopez Vicario, Antoni Morell, and Xavier Vilajosana. Addressing Mobility in RPL with Position Assisted Metrics. *IEEE Sensors Journal*, 16(7):2151–2161, 2016.

[5] Maha Bouaziz, Abderrezak Rachedi, and Abdelfettah Belghith. EKF-MRPL: Advanced mobility support routing protocol for internet of mobile things: Movement prediction approach. *Future Generation Computer Systems*, 93:822–832, 2019.

[6] Jihong Park, Ki Hyung Kim, and Kangseok Kim. An algorithm for timely transmission of solicitation messages in RPL for energy-efficient node mobility. *Sensors (Switzerland)*, 17:1–21, 2017.

[7] Olivier Cappe, Simon J. Godsill, and Eric Moulines. An overview of existing methods and recent advances in sequential Monte Carlo. *Proceedings of the IEEE*, 95(5):899–924, 2007.

[8] Hossein Fotouhi, Maryam Vahabi, Iliar Rabet, Mats Björkman, and Mário Alves. MobiFog: Mobility Management Framework for Fog-assisted IoT Networks. In *IEEE Global Conference on Internet of Things GCIoT'19, 04 Dec 2019, Dubai, United Arab Emirates*, 2019.

[9] Markus Becker, Koojana Kuladinithi, and Carmelita Görg. Modelling and simulating the trickle algorithm. In *International Conference on Mobile Networks and Management*, pages 135–144. Springer, 2011.

Chapter 7

Paper C:
Pushing IoT Mobility Management to *the Edge*: Granting RPL Accurate Localization and Routing

Iliar Rabet, Shunmuga Priyan Selvaraju, Mohammad Hassan Adeli, Hossein Fotouhi, Ali Balador, Maryam Vahabi, Mário Alves, Mats Björkman
At the IEEE 7th World Forum on Internet of Things, July 2021

Abstract

Accurate and timely mobility support in Internet of Things (IoT) applications is a challenging issue, considering the inherent scarce resources of IoT devices. However, the computational, memory and communication burden may be pushed into more "muscled" *Software Defined Network* (SDN) controllers. A centralised controller can exploit its global view of the network to predict and support seamless handovers. However, it requires the controller to be enhanced with extra link quality information. In this work, we present SDMob, an SDN-based mobility management solution that lifts the burden of computation intensive filtering algorithms from resource constrained nodes and achieves accurate and fast handovers upon nodes' mobility under *Routing Protocol for Lossy Low-power Networks* (RPL) and *IPv6 over Low-Power Wireless Personal Area Networks* (6LoWPAN). We show that SDMob improves the baseline RPL and the state-of-the-art mRPL in terms of packet delivery ratio leveraging more reliable routing and applying *Particle* filter and variations of *Kalman* filter on radio signal strength data enables more accurate localization for complex real world trajectories.

7.1 Introduction

There is an increasing demand for mobility support in IoT applications such as in healthcare, industrial automation and environmental monitoring. Nevertheless, the *de facto* protocol stack for low power and lossy networks (LLN) - RPL/6LoWPAN have not been designed for coping with a dynamic topology. In fact, they cannot provide a timely and accurate response to constant and fast topological changes in the network.

It has been shown that different mobility models affect the behavior of RPL in distinct ways [1], but it is believed that the baseline RPL protocol has proven to degrade quality-of-service upon mobility [2]. In the standard RPL, the Trickle algorithm is responsible for adapting the transmission frequency of control packets to the rate of changes in the topology. Increasing control packet's rate could result in better resiliency against mobility, but at the cost of higher communication and energy overheads. Nevertheless, since no predictive measure is taken, mobile node routes are only updated after disconnection, leading to network inaccessibility periods that will cause packet loss/delay.

A proactive approach to support seamless handoff could rely on Bayesian filters (such as a Kalman filters) or other predictive data processing techniques to forecast the future localization of mobile nodes. Here, a filter is referred to the methods that estimate the state of a temporal variable, which is usually observed under noisy measurements [3].

It is common to have *a priori* knowledge of the number of static nodes (in fixed positions) and the mobile nodes being assisted by an *Inertial Measurement Unit* (IMU). In such a scenario, it is practical to exploit statistical Bayesian prediction models (like Kalman and Particle filters) to fuse these two sources of information and, thereby, benefit from an accurate localization which leads to improvement in network responsiveness.

Kalman filter is proved to be unbiased (average error across all the recursive runs is zero), consistent (filter is neither overconfident nor under-confident) and optimal (minimum estimation error) [3]. However, in Kalman filter the posterior distribution (after the observations) can be computed in closed form only when the relation between states and observations is linear and the measurement and prediction noise follow a Gaussian distribution [4]. To tackle the nonlinear system models, *Extended Kalman Filters* (EKF) may be preferred; they use *Taylor series* to linearize the equations, trading for a negligible approximation error. On the other hand, Unscented Kalman Filter (UKF) and Particle filters have shown higher accuracy in prediction with bi-modal distribution of the error [3].

Implementing accurate predictive models may require higher computation capacity than mainstream IoT devices can afford. Resource-constrained nodes can hardly support lightweight filters (such as Kalman filters), provided that the position of the static nodes is hardcoded (in the mobile node). These limitations can be alleviated by offloading the computation burden to some external entity, such as a Software Defined Networking (SDN) controller. This can raise many challenges since the extra control packets between the SDN controller and the nodes lead to an extra traffic load. In this paper, we propose SDMob, an SDN-based mobility management architecture that relies on simple yet accurate localization mechanisms running in the SDN controller.

The main contributions of this work are listed below:

- Design of an SDN-based mobility management architecture –dubbed as SDMob– for seamless, reliable and timely mobility support.
- Implementation and fine tuning two filters (Particle filter and UKF) to determine mobile node position within a non-linear trajectory to enhance predictive routing.
- Implementation, integration and evaluation of the SDMob architecture into the RPL/6LoWPAN protocol, over a Contiki/COOJA + Linux ecosystem, comparing it against a benchmark non-SDN-based mobility solution (mRPL [5]).

This paper builds on our previous work [6], where we provided an analytical model of the proactive handoff mechanism using a Particle Filter. The model demonstrated how the expected probability of packet loss decreases with the seamless handoff managed by the controller.

The rest of the paper is organized as follows: Section 7.2 provides a brief description of the limitations of RPL upon mobility, and outlines efforts to improve this behaviour, namely through SDN-based IoT network/mobility management frameworks; it also sheds some light on the benefits of using Bayesian filters for improving location estimation and handoff decisions. Section 7.3 describes the SDMob architecture and the used filters. Section 7.4 shows the details of SDMob implementation and test environment. Moreover, SDMob comparison with mRPL will be shown and discussed. Finally, in Section 7.5 we conclude the paper.

7.2 Related Work

Mobility-aware RPL routing. Mobility management can be performed at different layers of the protocol stack. There is substantial body of research exploring detecting of radio link failure and network disconnection in IPv6 Neighbor Discovery or in the MAC layer. Nevertheless, to avoid routing loops, mobility management should also be considered at the routing layer [7].

RPL is considered as the *de facto* routing protocol for IoT. While it naturally supports joining and leaving of nodes, it performs poorly upon the dynamics imposed to the network topology. RPL maintains a distributed data structure named *Destination Oriented Directed Acyclic Graph* (DODAG). The process starts with the root transmitting a DODAG *Information Object* (DIO) that embeds the needed information to construct a routing tree towards the root.

RPL allows two modes of operation –*storing* and *non-storing*– for downstream traffic. In non-storing mode, it is only the root that maintains the downward routes. This mode scales better since the memory footprint at intermediate nodes does not increase with the size of network. It should be pointed out that in RPL it is more challenging to support mobility for downstream traffic since a mobile node must notify the root (rather than only updating its parent for upstream traffic).

The authors in [7] classify RPL enhancements to support mobility into scenarios with networks including only mobile nodes (e.g. VANETs) and networks with both static and mobile nodes. For the former, the recommendations of the RPL standard to not setting the mobile nodes as routers cannot be respected. In this case, Tian et. al [8] try to adjust the Trickle timer according to mobile nodes' velocity and utilize geographical information as RPL metric. In case the mobile node is not equipped with IMU sensors, it is possible to estimate its position through the *Doppler Effect*, as explained in [9]. The most suitable model to predict the handoff depends on the available sources of data. For instance, a *Deep Learning*-based Long Short Term Memory (LSTM) model could be preferred, if a supervised data set exists for the model to learn from [10].

Location prediction models for proactive handoff. In [11], authors have proposed Kalman RPL, in which a mobile node transmits a beacon that includes its velocity information in specific intervals. After a positioning phase that estimates the current position of the mobile node using three static nodes in its vicinity, it can predict the future position of the node.

EKF-RPL [12] takes a similar approach but it employs EKF within RPL to support non-linear trajectories. There are also some efforts on adopting *on-demand* routing strategies when a node starts searching for a route for transmitting data. The *Lightweight On-Demand Adhoc Distance-vector* routing protocol - Next Generation (LOADng) [13] is one such protocol specifically designed to support any-to-any communication in LLNs, although it is not as well-studied as RPL. EKF-LOADng [14] predicts RSSI after a trilateration positioning. In the triangulation phase, a mobile node broadcasts a message asking for packets from its static neighbors. Responses from static nodes experience a random waiting time to avoid collision. Another challenge is that the mobile node has to be programmed with the position information of the anchors and perform the sophisticated filter on its own.

Particle filter leads to more accurate results and better resiliency against nonlinear moving trajectory and non-Gaussian noise compared to EKF [3]. Particle filter also known as *Sequential Monte Carlo* uses hundreds to thousands of samples to predict the future state and fuse the measurement, hence requiring a higher computation capacity than most constrained IoT nodes can provide.

Particle filter has been extensively used to support mobility in Unmanned Aerial Vehicle (UAV) assisted networks [15] or cellular communications as in [16] which also proposes a *Rao-Blackwellised* particle filter as a lightweight alternative to the baseline particle filter. There are fundamental differences between cellular networks and LLNs such as density of the network deployment, range of transmission and speed of mobile nodes.

SDN-enabled IoT network architectures. There is a growing popularity in using SDN-enabled solutions in IoT networks. It is too expensive in terms of to simply integrate the common SDN solutions and standards within constrained IoT networks without re-designing the SDN to consider IoT limitations [17]. Therefore, there is a requirement for devising solutions targeting IoT networks with reduced complexity and operational cost.

Efforts have been made (including by standardization bodies) to design solutions for managing IoT network. The *Internet Engineering Task Force* (IETF) has a recent draft for infusing data routes into the network that is called *DAO projection* [18]. It defines a framework for the root node to initialize some options in DODAG *Advertisement Objects* (DAO) through new control messages, namely *Project DAO Request* (PDR) and *PDR-Acknowledgement* (PDR-ACK). This enables the root node to install routes in either the source or intermediate nodes along the path. The mechanism is a low-overhead substitute for implementing centralized network management in IoT networks.

Coral SDN [19] is another RPL-based solutions which allows an SDN controller to manipulate RPL routing parameters such as the interval used by the Trickle timer. The interval is the duration between successive DIS messages from a leaf node, which is an important configuration to adapt the responsiveness of the network.

MobiFog [20] is our previous work on centralised mobility management, where the discovery of alternative parents is performed using the actual data packets instead of dedicated control packets (beacons).

Overall, the literature in IoT networks mostly neglect more sophisticated and computation-intensive filters for network/mobility management, as well as SDN-based architectures. We believe that SDMob paves the ground for employing more accurate filter/localization algorithms at the SDN controller, towards improved performance upon mobility in IoT networks.

7.3 SDMob Architecture

In this section, we present the proposed SDMob architecture in two subsections. First we describe the filter/localization process. Then, we outline the SDMob architecture with the mobility management mechanism.

7.3.1 Filter design

Filters help with the prediction of future position of Mobile Node (MN), based on radio signal strength data and velocity. A more accurate localization of the MN improves network connectivity through proactive handoff. Filters model the states (positions) and observations as a *Hidden Markov Model* like the one illustrated in Figure 7.1. Within the model, the *Markovian* property holds true, meaning that each state (k-th) at a given time only depends on the state before ($k-1$-th) and the states can be estimated not directly but through some noisy observations.

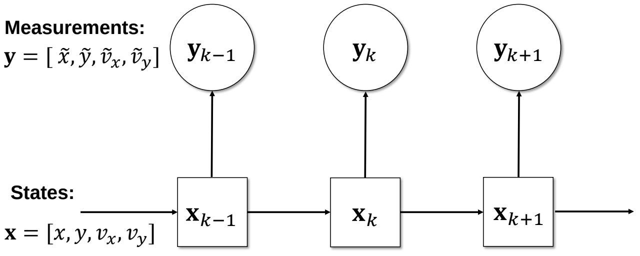


Figure 7.1: Markovian dependencies for the tracking problem

The state vector and measurement vector at k-th time step are $\mathbf{x}_k = [x \quad y \quad v_x \quad v_y]^T$ and $\mathbf{y}_k = [\tilde{x} \quad \tilde{y} \quad \tilde{v}_x \quad \tilde{v}_y]^T$ respectively. x and y are positions within Cartesian coordinates. Measurements are based on the Received Signal Strength Indication (RSSI) and IMU observations. Using the path loss model in Equation 7.1, the controller estimates the distance between mobile nodes and three distinct static nodes and then applies triangulation. In Equation 7.1, P_1 denotes the received signal strength in a 1 meter distance and α is a constant describing the radio propagation in the environment [21].

$$d = 10^{(\mathrm{RSSI}-\mathrm{P}_1)/10\alpha} \tag{7.1}$$

The state vector can be related to the previous state (Markov Property) using the Equation 7.2 and relation of each state with the corresponding measurements can be described using Equation 7.3. The estimated probability of the current state is based on the observations up to the current state, formulated as $p(x_k|y_0, y_1, \ldots, y_k)$. Then in the prediction step the probability density $p(x_{k+1}|y_0, y_1, \ldots, y_k)$ is computed which is the probability density of the next state $k+1$ knowing the observations as of the current time step.

$$\mathbf{x}_k = F(\mathbf{x}_{k-1}) + \mathbf{n}_{k-1}, \tag{7.2}$$

$$\mathbf{y}_k = H(\mathbf{x}_k) + \mathbf{r}_k, \tag{7.3}$$

In this system model, F is the motion model function that describes the relationship between states in time, and H is the function that relates the current state to the noisy observations at the current time instance. $\mathbf{n}_k$ and $\mathbf{r}_k$ are the prediction and measurement noise respectively. These two noises are mutually independent. The aforementioned functions and noises determine applicability of the classic filters. Kalman Filter is only advantageous in linear functions and Gaussian noise. Some enhancements such as *Enhanced Kalman Filter* (EKF) focus on handling non-linear F and H functions. However, to counteract non-Gaussian noise under a non-linear trajectory in the controller, we are compelled to adopt some other techniques such as UKF or particle filter.

Unscented Kalman Filter Instead of solving the intractable non-linear equations, UKF picks a number of deterministic samples that can be processed by the non-linear functions easily. These samples or *Sigma Points* and their corresponding weights satisfy some conditions defined by the Unscented Transform. Then these sigma points are mapped by a non-linear function to the new points. Finally, a good estimate of posterior mean and covariance of the transformed points is calculated using simple weighted averaging. We have implemented both filters using the python libraries introduced by [22].

Particle Filter Particle filter also known as Monte Carlo Sampling maintains a set of fully random particles, though the number of samples is expected to be much higher. The filter can take advantage of any a priori knowledge of the obstacles and infeasible positions when initializing the samples. Once it receives the RSSI measurements it updates their weights and based on the

IMU information moves all the particles together. On the other hand, particle filter has the feature of defining obstacles or feasible areas for the mobile node by simply removing the samples that get to be outside the legit area. In long term some of the samples can turn out to be irrelevant so particle filter can disregard those improbable particles and perform a re-sampling mechanism. When the number of effective samples (their weight crosses some threshold), the filter can take different re-sampling strategies. A systematic re-sampling in which the new samples are scattered around the state space is used in our implementation.

7.3.2 SDMob architecture

Figure 7.2 details the SDMob architecture, including an illustrated topology and the underlying technologies. Within WSN, network is composed of Mobile Node (MN) and Static Node (SN). The border-router utilizes the Serial Line Internet Protocol (SLIP) at border router between WSN and external SDN-Controller. SLIP converts the radio messages to a standard for serial links and vice versa, but it involves some unavoidable delay due to serialization. To interconnect the controller to the border router, we use Linux kernel pipes.

Chiefly, SNs act as repeaters or forwarders for MNs, and aid in their localization. We make two assumptions of the WSN: (i) implementation of RPL/6LowPAN protocols exists in the SNs and (ii) MNs are equipped with IMU sensors. In this design, software-defined network management supplements existing RPL/6LoWPAN with programmable network through route updating anchors which enables logically seamless connection for mobile nodes. These functionalities could be implemented either in the application layer on top of the protocol stack or integrated in the of RPL/6LoWPAN. Beacons sent from MN and forwarded by SNs towards SDN controller. Based on the observations made by beacons, controller provides route updates. The handover process is carried out in 3 steps, as illustrated in Figure 7.2.

- Step 1: MN broadcasts control beacon in a upstream with velocity information from IMU sensor.
- Step 2: All SNs in vicinity of MN receive and relay beacons with appended RSSI values in upstream towards the controller.
- Step 3: Controller runs the filter selects the best SN to act as new best parent, which is transmitted in a downstream packet towards SNs. Thereafter, only the best node relays the data packets generated by MN.

Seamless hand-off mechanism relies on a robust connection with the controller. Hence, low-complex design characteristics has to be involved in implementation of SDN architecture for WSN, which we detail below:

- **Avoidance of collision between control and data packets.** A centralized SDN-based controller introduces an additional control overhead. This increases the traffic through basic MAC-layer implementation of WSN, which is incapable of handling it and eventually more packet drops are experienced due to collisions. To streamline the traffic, a reserved period for control packets called *Control Window* (CW) has been implemented. CW can be adjusted based on network dynamics.

- **Configuring MN as RPL Aware Leaf (RAL).** As defined by a recent standardization effort to employ RALs in RPL [23], a leaf in RPL is a host that does not participate in further advertising the DODAG and relies on the RPL routers to forward its traffic. SDMob takes advantage of RALs to avoid excessive DIO packets as it is a major source of energy consumption when there is frequent topological changes. Another upside is reduced memory footprint in the MN. Second, this limits possibility of distinguishing MN as an intermediate node for other SNs. Last but not least, it would suffice to only notify the SNs of the current *best parent* rather than using the links to the MN that are much less reliable.

- **Sophistication of downward routing.** In standard RPL, upstream data transmission is favored as it is the most predominant traffic pattern in IoT domain. This extends to many mobility enhancements made to RPL as they also have weaker behavior in terms of downward traffic towards the mobile node. Handovers are treated locally without briefing the root node SDMob works with the non-storing mode of RPL which gathers more routing information at the root and uses source routing for downward data packets but builds upon the extra localization to also improve downward traffic towards the MN.

In Figure 7.3(a), a timeline demonstration of overall handoff process in SDMob is illustrated. In the data window, MN broadcasts data packets and selected SN or best parent forwards them to the border router. In the control window, MN only transmits the localization beacon and all static nodes receiving the control packet will append the measured RSSI to forward the packet to towards controller with CSMA-based unicast packets. The controller runs the filter and announces the new *best parent*. The old best parent stops forwarding data once it is notified of the SDN's most recent choice. After the CW, data transmission is resumed. Figure 7.3(b) shows a timeline of hand-off

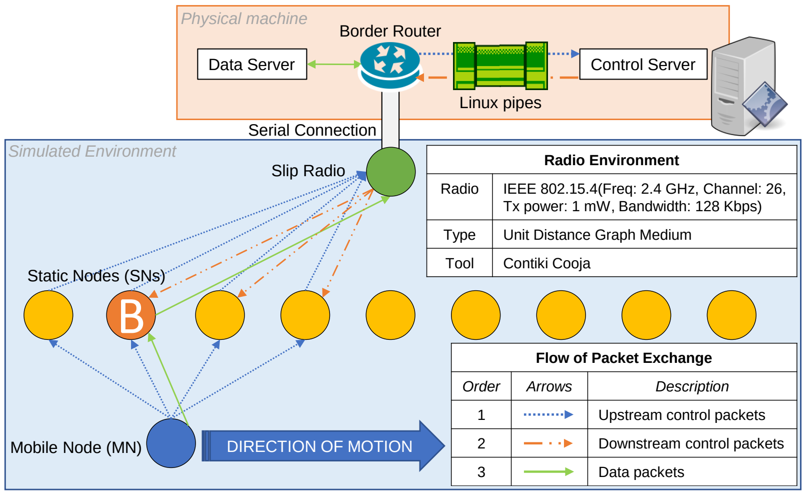


Figure 7.2: SDMob architecture schematic with exchange flow: i) MN broadcasts beacons in upstream towards controller through SN; ii) controller broadcasts the new *best parent* in downstream to SNs; and iii) data communication. Only B (best parent) handles data packets generated by MN.

mechanism in benchmark mRPL. In mRPL, the Mobile Node (MN) operates in two phases. In the *Data Transmission phase*, the APs constantly monitor the link quality and compare the RSSI value to a threshold T_l. If the link quality degrades, the APs notify the MN with a beacon and it stops transmitting data packets and instead it will start the *Discovery Phase* by sending DIS packets to ask the APs to respond with DIO packets. Then the mobile node analyses the received DIO packets and if the RSSI values from other APs turn out to be higher than T_l it performs the hand-off and resumes data transmission, otherwise it continues sending DIS packets.

7.4 Simulation setup and analysis

SDN controller has been implemented using Linux machine with Python-based filters which connects to C-based Contiki border router. For the IoT RPL/6LoWPAN network, we rely on the Contiki-NG/COOJA simulation environment [24]. Contiki-NG is an open-source embedded operating system which is easily portable to commodity hardware.

Simulation setup. We compare the performance of SDMob with filters (particle filter and UKF) with mRPL as well as with the default RPL. We consider two different trajectories first a linear and second a circular trajec-

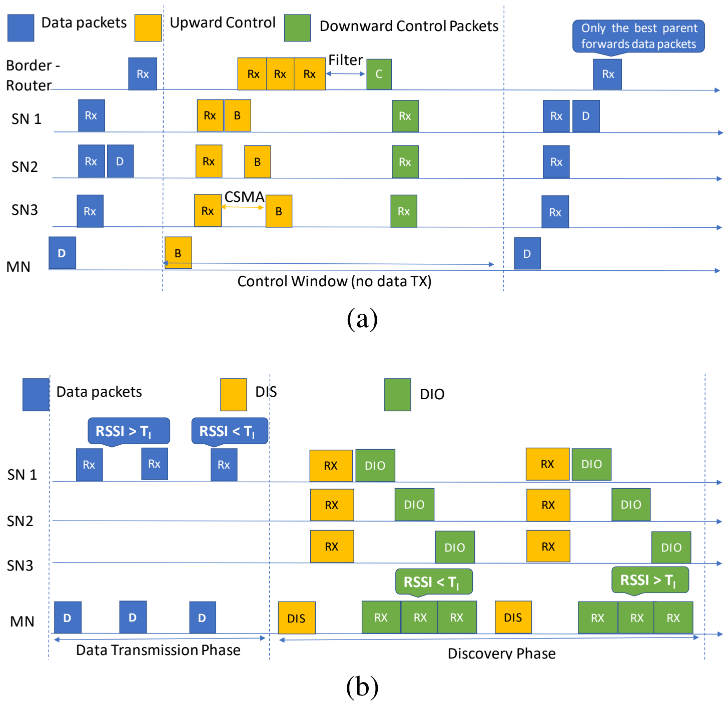


Figure 7.3: A timeline diagram of handoff in (a) SDMob (b) and mRPL.

tory. The architecture allows multiple mobile nodes given that controller can run different instances of the filter and differentiate beacons as they include MN's IP address. Though due to space limitations, we share the results regarding a single mobile node with different moving tracks. The simulations consider different sampling intervals for data transmission (here the mobile node sends data packets periodically), path-loss variance and CW. We have employed mRPL [5] as the benchmark since it provides a mobility solution for IoT networks using a non-SDN-based framework. We could not find any available implementation of those related work with variations of Kalman Filter. The experimental results considers three main performance metrics, including (i) packet delivery ratio (PDR) to measure reliability, (ii) End-to-End (E2E) delay, and (iii) Root Mean Square Error (RMSE) to measure positioning accuracy . We investigate the impact of different parameters on these metrics as follows:

Impact of data transmission interval. Data transmission interval is the time interval between consecutive transmission of data packets. As shown in Figure 7.4(a), we analyzed the E2E delay for data packets. We observed that

E2E delay is in the range of 150 ms with SDMob, while mRPL has much lower delay (≤ 50 ms), slightly less that RPL's ≈ 100 ms. This is important to note that delay in RPL is calculated only for packets that have been successfully transmitted ($\approx 10\%$). This means that packets in RPL with mobile node is either transmitted before parent switching or getting lost due to the slow parent switching process and lack of preferred parent. Additional delay in the SDMob can be explained due to:

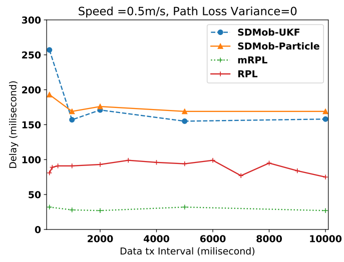
- CW occupies a specific time interval for control packets, which postpones the transmission of data packets.
- Serialization delay occurred by SLIP protocol to convey radio messages to the Linux-based controller. This delay includes 20ms polling intervals as well as the processing time.

Unlike mRPL which performs handoff based on the most recent RSSI averages, performance of handoff in SDMob is independent of the network traffic – see Figure 7.4(b). This robustness and resiliency is achieved at the cost of a constant control overhead and the incurred delay. For short data transmission intervals, mRPL provides about 80% PDR with a decreasing trend for longer intervals surpassing standard RPL's poor PDR ($\approx 20\%$). **SDMob constantly outperforms mRPL and RPL with PDR above 95% across different data rates**.

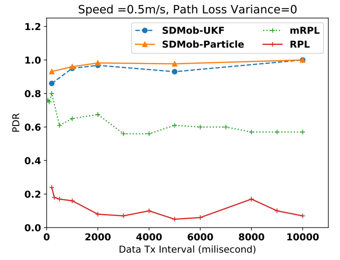
Impact of various filtering methods and trajectories. To analyze the positioning accuracy of filters, RMSE metric is mostly used in the literature (compared to the average error). The reason is that RMSE assigns a higher penalty to large errors. As it is shown in Figure 7.4(c), **particle filter provides a better positioning error compared to the UKF for both linear and circular trajectories with different sampling rates**. Increasing the physical speed of mobile node also deteriorated the metrics as the number of handoffs increases.

Experiments also revealed that SDMob is more resilient even under path loss variance (not illustrated in the figures) and provides better PDR. This can be explained by the fusion of measurements in the filter that takes advantage of IMU information in parallel with the non-stable RSSI observations.

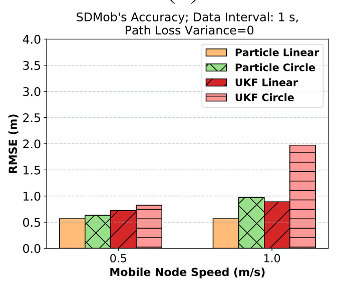
Impact of CW size. Longer control window increases the stability of the network through longer network monitoring, which thereby increases the probability of a successful transmission, though it imposes a longer delay on data packets. As expected, the simulations exhibit an increasing trend both in PDR and delay while the CW length increases. The CW only needs to cover the time for a round trip time for control packets. The average delay for control



(a)



(b)



(c)

Figure 7.4: Simulation results showing (a) E2E delay and (b) PDR for different sampling rates with CW:100 ms and linear trajectory and (c) positioning accuracy for different trajectories and velocity.

Table 7.1: Comparing memory footprint of SDMob with Contiki-NG's default RPL implementation.

Program	SD-Mob		mRPL		Contiki-NG base	
	ROM	RAM	ROM	RAM	ROM	RAM
MN	46 kB	7628	44 kB	7898	43 kB	7394
Data Server	163 kB	7400	46 kB	7916	43 kB	7348
SN	44 kB	7632	45 kB	7928	44 kB	7632

packets in a two-hop network including the delays for SDMob's particle filter algorithm is about 200 ms. For CW ranging from 0 to 300 ms, we observed average delays from 175 to 220 ms, while PDR increased linearly from 89 to 100 percent.

Memory footprint. Memory consumption of the nodes is a measure that can testify to offloading excessive computation to the SDN controller. Contiki-NG's implementation of RPL has optimized its code and since SDMob is based on Contiki-NG it is unfair to compare it with mRPL, which is based on Contiki 2.6. As shown in Table 7.1, for the mobile node and the anchor nodes, SD-Mob's MN and SN require about 3% less RAM memory compared to mRPL and not too much overhead compared to base Contiki-NG (no mobility support). The server's memory consumption is much higher than mRPL but since SDMob's server is offloaded to the Linux machine, it will have no negative impact in the IoT network. So overall, we can argue that SDMob has outsourced the mobility related computations.

7.5 Conclusion

In this paper, we have addressed the design and implementation of SDMob, an SDN-based mobility solution for IoT RPL/6LoWPAN networks. The proposed architecture is based on an applying filters to radio signal strength and velocity measurements captured by the anchor nodes. This mechanism enables a more accurate prediction of the mobile nodes and consequently a more precise selection of the best anchor nodes. Simulation results showed that by using a periodic beaconing mechanism, SDMob's PDR and delay are independent of the network traffic, while mRPL is tightly coupled with data transmission interval. RPL has shown to be very low responsive to dynamics in the network, leading to high packet losses. The CW mechanism and the extra control packets imposed an overhead that justifies the higher delay in SDMob. Since real-world environments exhibit more varying link behavior, future experiments will be based on a real hardware setup. Further tests on scalability of the system, impact of node density, number of mobile nodes, trajectory of movement and dynamic adaptation of CW are also envisaged.

We plan to further extend this work to support adaptive beacon rates, automatically detect radio characteristics of the environment and evaluate the scalability in terms of number of mobile nodes and density of static nodes in our future work.

Bibliography

[1] Bardia Safaei, Aliasghar Mohammadsalehi, Kimia Talaei Khoosani, Saba Zarbaf, Amir Mahdi Hosseini Monazzah, Farzad Samie, Lars Bauer, Jörg Henkel, and Alireza Ejlali. Impacts of mobility models on rpl-based mobile iot infrastructures: An evaluative comparison and survey. *IEEE Access*, 8:167779–167829, 2020.

[2] Kevin C Lee, Raghuram Sudhaakar, Jianxia Ning, Lillian Dai, Sateesh Addepalli, JP Vasseur, and Mario Gerla. A comprehensive evaluation of RPL under mobility. *International Journal of vehicular technology*, 2012, 2012.

[3] Simo Särkkä. *Bayesian filtering and smoothing*, volume 3. Cambridge University Press, 2013.

[4] Olivier Cappe, Simon J. Godsill, and Eric Moulines. An overview of existing methods and recent advances in sequential Monte Carlo. *Proceedings of the IEEE*, 95(5):899–924, 2007.

[5] Hossein Fotouhi, Daniel Moreira, and Mário Alves. mRPL: Boosting mobility in the Internet of Things. *Ad Hoc Networks*, 26:17–35, 2015.

[6] Iliar Rabet, Shunmuga Priyan Selvaraju, Hossein Fotouhi, Maryam Vahabi, and Mats Bjorkman. Poster: Particle Filter for Handoff Prediction in SDN-based IoT Networks. In *EWSN 2020*, 2019.

[7] Patrick Olivier Kamgueu, Emmanuel Nataf, Thomas Djotio Ndie, Patrick Olivier Kamgueu, Emmanuel Nataf, Thomas Djotio, and Ndie Survey. Survey on RPL enhancements : a focus on topology , security and mobility To cite this version : HAL Id : hal-01713247. *Computer Communications*, 2018.

[8] Bin Tian, Kun Mean Hou, Hongling Shi, Xing Liu, Xunxing Diao, Jianjin Li, Yibo Chen, and Jean-Pierre Chanet. Application of modified

RPL under VANET-WSN communication architecture. In *2013 international conference on computational and information sciences*, pages 1467–1470. IEEE, 2013.

[9] Jihong Park, Ki Hyung Kim, and Kangseok Kim. An algorithm for timely transmission of solicitation messages in RPL for energy-efficient node mobility. *Sensors (Switzerland)*, 17:1–21, 2017.

[10] Chaoyun Zhang, Paul Patras, and Hamed Haddadi. Deep Learning in Mobile and Wireless Networking: A Survey. *IEEE Communications Surveys and Tutorials*, 21(3):2224–2287, 2019.

[11] Marc Barcelo, Alejandro Correa, Jose Lopez Vicario, Antoni Morell, and Xavier Vilajosana. Addressing Mobility in RPL with Position Assisted Metrics. *IEEE Sensors Journal*, 16(7):2151–2161, 2016.

[12] Maha Bouaziz, Abderrezak Rachedi, and Abdelfettah Belghith. EKF-MRPL: Advanced mobility support routing protocol for internet of mobile things: Movement prediction approach. *Future Generation Computer Systems*, 93:822–832, 2019.

[13] Thomas Clausen, Jiazi Yi, and Axel Colin De Verdiere. LOADng: Towards aodv version 2. In *2012 IEEE Vehicular Technology Conference (VTC Fall)*, pages 1–5. IEEE, 2012.

[14] Allan J.R. Gonçalves, Ricardo A.L. Rabêlo, Joel J.P.C. Rodrigues, and Luís M.L. Oliveira. A mobility solution for low power and lossy networks using the LOADng protocol. *Transactions on Emerging Telecommunications Technologies*, 31(November 2019):1–24, 2020.

[15] Baohua Shao and Mark S Leeson. PaFiR: Particle Filter Routing–A predictive relaying scheme for UAV-assisted IoT communications in future innovated networks. *Internet of Things*, page 100077, 2019.

[16] L. Mihaylova, D. Angelova, S. Honary, D. R. Bull, C. N. Canagarajah, and B. Ristic. Mobility tracking in cellular networks using particle filtering. *IEEE Transactions on Wireless Communications*, 6(10):3589–3599, 2007.

[17] Zohaib Latif, Kashif Sharif, Fan Li, Md Monjurul Karim, Sujit Biswas, and Yu Wang. A comprehensive survey of interface protocols for software defined networks. *Journal of Network and Computer Applications*, 156:102563, 2020.

[18] Pascal Thubert, R.A. Jadhav, and Gillmorem M. Root initiated routing state in RPL draft-ietf-roll-dao-projection-09, 2019.

[19] George Violettas, Sophia Petridou, and Lefteris Mamatas. Routing under Heterogeneity and Mobility for the Internet of Things: A Centralized Control Approach. In *2018 IEEE Global Communications Conference, GLOBECOM 2018 - Proceedings*, 2018.

[20] Hossein Fotouhi, Maryam Vahabi, Iliar Rabet, Mats Björkman, and Mário Alves. MobiFog: Mobility Management Framework for Fog-assisted IoT Networks. In *IEEE Global Conference on Internet of Things GCIoT'19, 04 Dec 2019, Dubai, United Arab Emirates*, 2019.

[21] Marco Zuniga and Bhaskar Krishnamachari. Analyzing the transitional region in low power wireless links. In *2004 First Annual IEEE Communications Society Conference on Sensor and Ad Hoc Communications and Networks, 2004. IEEE SECON 2004.*, pages 517–526. IEEE, 2004.

[22] Roger Labbe. Kalman and bayesian filters in python, 2014. *URL https://github. com/rlabbe/Kalman-and-Bayesian-Filters-in-Python*, 2019.

[23] Pascal Thubert and M Richardson. Routing for RPL leaves. *Work in Progress, draft-thubert-roll-unaware-leaves-05*, 2018.

[24] Fredrik Österlind. A sensor network simulator for the contiki os. *SICS Research Report*, 2006.

Chapter 8

Paper D: RPL-RP: RPL with Route Projection for Transversal Routing

Iliar Rabet, Hossein Fotouhi, Maryam Vahabi, Mário Alves, Mats Björkman.
At the IEEE 7th World Forum on Internet of Things, July 2021

Abstract

Routing Protocol for Low-Power and Lossy Networks (RPL) as the most widely used routing protocol for constrained Internet of Things (IoT) devices optimizes the number of routing states that nodes maintain to minimize resource consumption. Given that the routes are optimized for data collection, this leads to selecting sub-optimal routes, particularly in case of east-west or "transversal" traffic. Additionally, RPL neglects interactions with a central entity in the network for monitoring or managing routes and enabling more flexibility and responsiveness to the system.

In this paper, we present *RPL with Route Projection (RPL-RP)* that enables collecting siblings' relations at the root node in order to inject routing states to the routers. This backward-compatible RPL extension still favors collection-based traffic patterns but it enriches the way routing protocol handles other flow directions. We address different advantages of RPL-RP in contrast to standard RPL and evaluate its overhead and improvements in terms of end-to-end delay, control overhead and packet delivery ratio. Overall, RPL-RP halves the end-to-end delay and increases network reliability by 5% while increasing network overhead by only 3%.

8.1 Introduction

The current developments in the field of IoT promise to expand the applications with traffic patterns that are more complex than simple data collection that traditional networks are designed for. Hence, bringing ease-of-management and flexibility to the underlying infrastructure on top of which they operate is of utmost importance. However, the common protocol stack implementations barely support such features.

Low-Power and Lossy Networks (LLNs) are key components of IoT and provide wireless communication between sensors and actuators. RPL [1] has long been adopted by LLNs for its energy efficiency and minimum resource requirements. LLNs are characterized by high loss and fluctuations in the links and applications that mandate low data rates and high scalability. Different IETF working groups have designed a stack of protocols including IPv6 over Low-power Wireless Personal Area Networks (6LoWPAN) that defines header compression for IPv6 on top of IEEE 802.15.4 Medium Access schemes. This protocol stack was not initially designed for ease-of-interaction with a central network manager since the nodes were independent entities running a distributed control plane.

RPL maintains a data structure named *Destination Oriented Directed Acyclic Graph* (DODAG) and the functionality starts with a root node transmitting a *DODAG Information Object* (DIO) packet. All the receiving nodes will choose the best parent towards the root based on an objective function and will transmit new DIO packets to further increase the range of DODAG. The upward flow from the nodes to the root can start once a DIO packet is received but for downward communication, the root needs to wait to receive a *Destination Advertisement Object* (DAO) packet.

There are some limitations inherent to the design of the RPL that are biased to the upward collection-based traffic and all the other traffic directions are restricted to transmit only along the DAG resulting in "stretched" routes. For downward communication or any-to-any communication, the nodes in RPL are programmed to be either in *Storing* or *Non-Storing* mode, where the former requires more memory for keeping track of the nodes in the sub-trees of each node. For routing between two non-storing nodes in the network, the packets have to be transmitted all the way up to the root and back down to the destination. Even in the storing mode, they have to find a common ancestor, which is not necessarily the optimal path. Figure 8.1 exhibits how different modes of RPL stretch the routes while siblings can promote point-to-point routing. Besides the routing mode, the common metrics used by the RPL's objective function prefer the routes that have a better upward connectivity while links

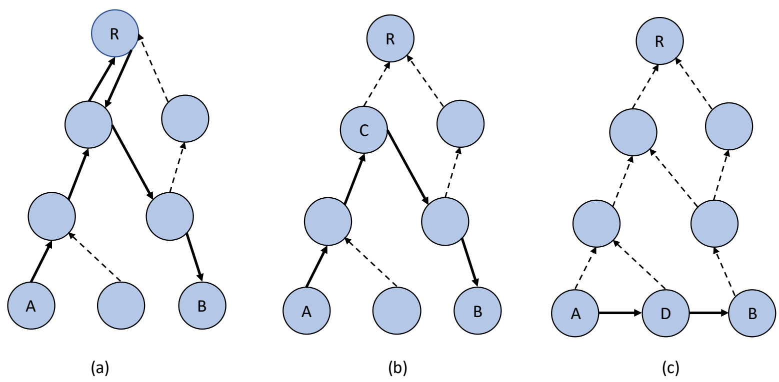


Figure 8.1: Examples of data transmission strategies for sending packets from node A to node B. (a) RPL's *non-storing* mode that requires passing through the root. (b) RPL's *storing mode* that permits passing through a common ancestor. (c) A shorter route including siblings.

can show asymmetric behavior in different directions [2].

Extensively computing all the routes for any-to-any communication does not abide by the resource constraints of the low-power nodes. A reasonable compromise is to have the root node injecting a handful of routing states into some of the in-network nodes when it perceives a partial yet sufficient knowledge of the network topology.

Another common limitation of RPL is the poor performance upon employing mobile nodes [3–5]. The Trickle algorithm [6] controls the frequency of transmission of RPL's DIO by trying to adapt its transmission rate to the level of stability in the network. Tuning more frequent transmission of DIO packets can update the routes in a timely manner but it will drain the battery power. In case a controller is aware of the real-time position of a mobile node, for example, a robot controlled by the devices at the edge, RPL has no standard way of manipulating the routing state in the nodes that are multiple hops away from the root.

A recent effort by IETF ROLL group [7] has also focused on the above-mentioned problems. The draft on the "root initiated routing state in RPL" defines new control packet types and control packet options that provide backward-compatible mechanisms for the RPL root to collect more information and install the so-called *projected routes* on the selected nodes. The projected routes are of a higher priority to the nodes.

In this paper, we present RPL-RP, an open-source implementation of the

RPL with route projection, over Contiki-NG operating system along with a dashboard for monitoring the links. We also analyze its applicability to some of the scenarios that are challenging to the traditional RPL.

The paper is organized as the following: Section 8.2 reviews the related work and outlines the gaps in the literature. In section 8.3, we explain some of the details of the implemented system and then evaluate it in terms of overheads and improvements in section 8.4. Finally, section 8.5 concludes the paper and outlines prominent research directions.

8.2 Related Work

A substantial body of knowledge belongs to centralized management in constrained networks. Sensor OpenFlow [8] is one of the earliest efforts for offloading the network control in order to bring about more flexibility against dynamic policies and ease of management. In a similar approach, SDN-WISE [9] defines its own Topology Discovery layer in the constrained nodes to enable cross-layer operations. An integration with the ONOS [10] controller, allows MAC layer scheduling as specified by TSCH mode of IEEE 802.15.4 and increases connectivity for mobile nodes [11].

In μSDN [12], the authors argue that a centralized controller needs to be compatible with legacy RPL nodes and introduces some optimizations that allow μSDN to overcome the constraints that are common to IoT nodes. Optimizations can include avoiding packet fragmentation, source-routed control packets, configuring update timers. Hydro [13] was another effort to improve the distributed DAG formation with centralized primitives.

Not all the efforts to support any-to-any routing in RPL use a centralized entity. Authors of ORPL [14] combine RPL with opportunistic routing which means that traffic can be forwarded to any node based on the information about its sub-tree. Bitmaps and bloom filter are used to represent this information in a compressed format to avoid memory overflow. Bacceli et al. [15] introduced on-demand mechanisms to discover routes based on flooding control packets. RFC6997 [16] documents a standardized version of P2P-RPL that defines a new operation mode in which the *Origin* creates a temporary DAG along the main DODAG.

In an earlier work [17], we introduced an extension to RPL in which a centralized entity monitors the link qualities for a mobile node and defines some thresholds and timers to update the routes accordingly. Such solutions can also enable the networks to benefit from the computation capacity in the edge nodes to implement tracking algorithms such as particle filter and Unscented

Kalman Filter to predict the future position of the mobile nodes and starting the handover process prior to link disconnection [18].

Thubert et. al [19] proposes a framework for an SDN-based TSCH scheduler that meets the requirements of deterministic networking. The authors claimed that the key to improving reliability and mitigating interference is diversity. Diversity can be achieved in different domains as spatial diversity is leveraged with multi-path routing, temporal diversity by re-transmissions, and frequency diversity using channel hopping.

The IETF draft on DAO projection [7] defines some primitives to involve the central border router in the distributed operation of RPL and classifies route projection into *Storing Mode Projected Route* (SMPR) and *Non-Storing Mode Projected Routes* (NMPR). The mode for projected routes is independent of RPL's operation mode, meaning that the network can consist of storing mode RPL working with non-storing route projection or vice versa. NMPR uses source routing for the data packets but in SMPR root node asks the source node to update the routing state in all the intermediate nodes. The ROLL working group is currently actively working on this document and to the best of our knowledge there is not any available implementation to compare with. In both modes of DAO projection, getting acknowledgement from either source or destination would suffice. We suggest a reform to put all the intermediate nodes in direct connection with the controller rather than getting an acknowledgement only from source or destination of the path. This will ease troubleshooting since the controller gets to know which link in the path is troublesome. The standard however does not specify how and which routes should be calculated.

8.3 Design and implementation of RPL-RP

In this section, we explain how RPL-RP extends the RPL protocol to fulfill the following requirements:

- Installing point-to-point routes to optimize the path length
- Collecting sibling information besides parent-child relations in the RPL root to be used in a topology viewer dashboard or a controller
- Designing real-time interaction with a manual or automatic controller.
- Reducing the routing header by eliminating the source routing header or loose source routing.

RPL-RP is supposed to provide routes along a track, which is an ordered set of addresses that data packets are supposed to go through. A track is formed

Track ID	Flags	Reserved	DAO Sequence	IPv6 address of Track	Options
1 Octet	1 Octet	1 Octet	1 Octet	16 Octets	VIO/SIO

VIO Packet Format

Type + length	Flags	Segment	SRH-LoRH	VIA Address 1	….	VIA Address n
2 Octet	1 Octet	3 Octet	2 Octet	2 to 16 bytes as indicated by compression type		

SIO Packet Format

Type + length	Flags	Opaque	Step of Rank	Sibling DODAG ID	Sibling Address
2 Octet	1 Octet(including Compression Type	1 Octet (For LQI possibly)	1 Octet	2 to 16 bytes as indicated by compression type	2 to 16 bytes as indicated by compression type

Figure 8.2: The packet format for DAO packet including VIO option for P-DAO or SIO for upward DAO packets.

to include a single source (track ingress) and destination (the track egress). It is maintained by a local instance of RPL and gets the IP address of the RPL instance as its track ID. A complex track can also be defined to append two or more segments. A track can be installed in the main (Global) instance of RPL to enable routes to the root with different objectives or within other instances of RPL, as for transversal routing.

The new control message types that are introduced are *Projected DAO* (P-DAO), *P-DAO Request* (PDR) and P-DAO-ACK. As the name suggests, PDR is used to ask the root to install the routes towards the track egress for a requested *lifetime*. It is usually sent by the source of the track. The controller responds by sending a sequence of P-DAO messages comprised of *Via Information Option* (VIO) that can be acknowledged using P-DAO-ACK. VIO is a new *Control Message Option* designed to be included in the P-DAO packets, which is a sequence of IPv6 addresses of possible next hops. Figure 8.2 illustrates the packet format for P-DAO, which is identical to the DAO, except for the VIO option. P-DAO packets carry exactly one VIO option.

Figure 8.3 presents a sequence diagram of the control packets that enable route projection. The root node initiates the process by sending DIO and consecutively other nodes broadcast their objective function and in turn DAO packets are sent to the root. At this stage, point-to-point routing is performed

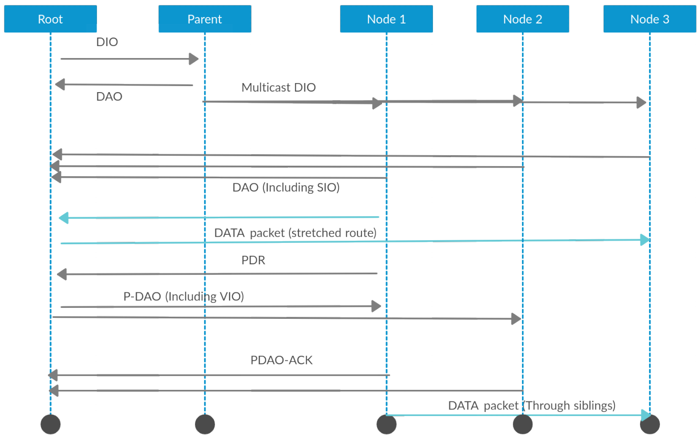


Figure 8.3: RPL converges to a set of routes that only include parents. Exchanging the P-DAO allows the in-network nodes to leverage more optimized paths not necessarily limited to parents.

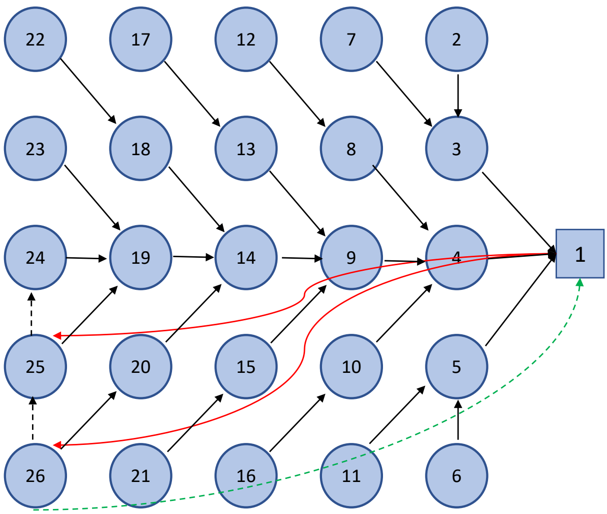
through the root node. Projected routes can only be installed after initial bootstrapping since they rely on the infrastructure that RPL provides. After exchanging the PDR and Projected DAO, the data packets can be disseminated through siblings.

To achieve low overhead in RPL-RP, routers store the IP addresses of their parents only, as their default route and point of attachment to the root node. This is on the ground that RPL was primarily designed for collecting data. For transversal routing, the traffic is delivered upward (to the root or a common ancestor in non-storing mode or storing mode respectively), and then downward towards the destination.

In RPL non-storing mode for point-to-point routing, data packets are equipped with a source routing header that contains the address for all the intermediate nodes in the path. Although this approach enlarges the routing header, intermediate nodes are not required to maintain the routing information and only the root node keeps the state of parent-child relationships.

However, in RPL storing mode, every single node in the path is stateful and needs to maintain consistency with other nodes and update data packet's next hop using the Hop-By-Hop option. Storing mode is often criticized for higher memory footprint and routing state inconsistency between the nodes. On the plus side, data packets only contain one IPv6 address in the header.

Projected routes also face a similar trade-off whether to choose source rout-



Node 25

Dest	Next Hop
24	24 (projected)
Default	19

Node 26

Dest	Next Hop
24	25 (projected)
Default	20

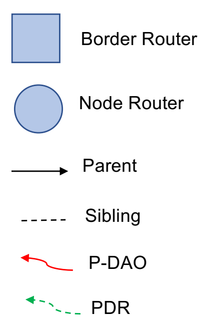


Figure 8.4: Simulation topology with 25 routers, where for a traffic from node 24 to node 26, the border router projects the routes after receiving the PDR packet from node 24, assuming that nodes have previously informed the border router of their siblings using SIO option in DAO packets.

ing or hop-by-hop routing header. Currently RPL-RP works with Hop-By-Hop option mode for data packets. The controller needs to send a P-DAO to all the hops in the track once it receives a PDR from track source. For example as illustrated in Figure 8.4, node 26 starts a flow to node 24 and transmits a PDR to the border router. In response border router establishes a connection to the nodes in the path (except for the track egress), rather than asking the track ingress to forward the P-DAO as in SMRP in [7]. Thereupon, nodes number 26 and 25 receive the P-DAO and install an entry for the projected routes in the routing table with a higher priority than the default upward routes.

On the plus side, sending the P-DAO to multiple nodes, allows the controller to become more resilient against link failures since it gets a separate confirmation from each hop and identifies the failed node in the track by getting P-DAO-ACK containing the error code. Additionally, the data packet's header does not expand with the number of hops. Another advantage is the constrained track ingress nodes are not required to implement the logic for handling the errors and installing the projected routes. Although this design is likely to marginally increase the number of P-DAO packets, our simulation results will show that the additional overhead is negligible.

To collect the extra information at the root node, another option is added to the DAO packets, namely *Sibling Information Option* (SIO). This option contains the IP address of the neighbors and the corresponding link qualities. The number of siblings can be huge, especially in dense networks. In that case, it is not possible to inform the root of all the siblings without an oversized DAO packet, and thus it requires limiting the number of addresses to be included in the SIO. A considerable future work is studying the process of selecting the siblings to include in the SIO option in a dense network or allowing fragmentation of big P-DAO packets. Based on the compression method that is indicated in the flags field for SIO or in the SRH-LoRH field for VIO, the addresses can be of different size from 2 to 16 bytes. Our measurements were based on full form of addresses in VIO, but due to the high number of siblings, we used 8 bytes compressed format in SIO. We defined a maximum number of 3 SIO options to avoid MAC layer fragmentation. The process of selecting the siblings is random. Most probably, choosing more siblings with lower Received Signal Strength Indicator (RSSI) will help in increasing the coverage of the network, though destabilizing the projected routes. It is important to note that devising a smart algorithm for selecting siblings is out of the scope of this paper.

The root node can be co-located with the controller but in this work, these nodes are separate entities, communicating through a JSON-based south-bound API. The root node parses the JSON file and sends P-DAO packets once it gets triggered through a web API or receives a PDR packet.

We make our source code and demo available for the sake of reproducibility[1].

8.4 Results and discussion

This section evaluates the performance of RPL-RP in terms of *end-to-end delay*, *memory footprint* and *communication overhead* incurred by the newly defined primitives. The simulation consists of a border router, a grid of 25 Tmote Sky nodes positioned as in 8.4 emulated in the Contiki/COOJA environment. Sky motes are deliberately chosen to show that DAO projection is capable to be implemented even using the old and low capacity motes although RPL-RP can also be implemented on almost any mote that supports RPL stack. The number of active point-to-point UDP flows is also controlled by the controller and increases over time. We gradually increase the number of flows until at least one transversal flow is running between all the nodes. At the MAC layer, we use CSMA and data packets are being sent every two seconds. The controller starts the transversal flows after the initial convergence. We show that with RPL-RP, a negligible overhead in the control traffic and a tolerable memory footprint can be traded for better latency and resiliency in the data plane.

RPL-RP benefits routing in different forms. First and foremost, it reduces the number of hops that data packets go through while lifting the burden of relaying congestion from the nodes that are closer to the root node and balances the load and energy consumption of the nodes. Second, by using the Hop-By-Hop option instead of Source Routing Header in downward routes, it reduces the header size for data packets. Last but not least, in case there is a local source of interference, RPL-RP's controller has the ability to inject convenient routes to bypass the lossy links.

Figure 8.5 demonstrates the end-to-end delay of data packets measured during the experiment. The end-to-end delay for data packets is proportional to the number of hops in the path. Other parameters such as link quality and number of re-transmissions also matter but RPL-RP accomplishes mostly through reducing the routing stretch. So performance of RPL-RP depends considerably on the topology. For instance, a very deep DODAG can significantly enjoy benefits of projected routes but a shallow network with long east west distance may not take advantage of it so much. In our specific grid topology (same as 8.4), the routing distance is scaled down from an average of 8 to 2 hops. **In RPL-RP, end-to-end delay is halved in high traffic scenarios compared with the default RPL.** RPL-RP can also tolerate increasing number of flows more smoothly and rate of increasing latency is lower compared to tra-

[1]`https://bit.ly/355DZbj`

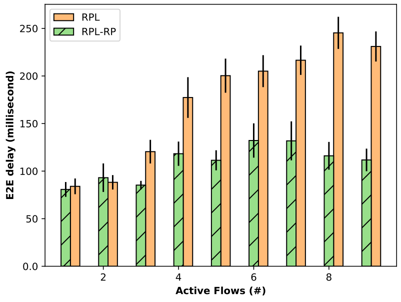


Figure 8.5: End-to-end delay of RPL-RP and RPL for different number of flows.

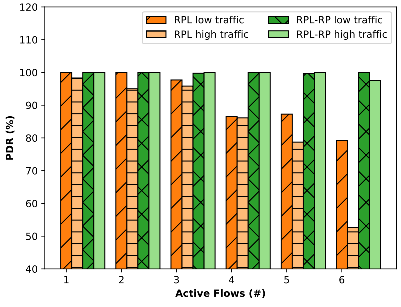


Figure 8.6: Packet Delivery Ratio for different number of flows.

ditional RPL. The confidence intervals indicate that network jitter follows the same trend as mean delay indicating another superiority of RPL-RP.

Besides latency, RPL-RP enhances the resiliency of the routing protocol against different causes of packet loss. In LLNs, it is very common to encounter packet losses due to the overflow in the packet queues specially as the closer nodes to the root get congested. In our tested scenario the loss rates of all the links are equal, thus reducing the path length would promote packet delivery ratio. As illustrated in Figure 8.6, RPL-RP reduces packet losses significantly both with increasing number of flows or with increasing traffic within the same flows. High traffic scenario comprised of 2 packets per second, which increased the loss rate up to 50 percent for RPL with 6 flows. In low traffic scenario nodes transmit 1 packet per second and RPL experienced around 80 percent delivery ratio.

RPL-RP provides nearly 100% packet delivery ratio regardless of network traffic due to bypassing the congested links closer to the root node.

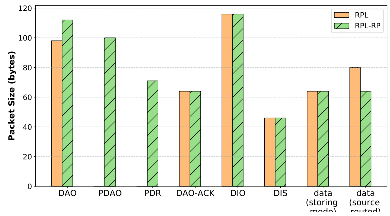


Figure 8.7: Comparing size of data and control packets in RPL with RPL-RPL.

The control traffic is generally governed by two factors: (i) the frequency of the transmissions and (ii) the packet size. To study how RPL-RP expands control traffic, it is important to determine how single packet types evolve within this protocol, then we explore the accumulated overhead. DIO and DODAG Information Solicitation (DIS) packets have the same characteristics for both RPL and RPL-RP, and thus they are excluded from the overhead calculation. Frequency of DAO packet transmission is also unaffected but since we append the new SIO option (with maximum 3 siblings), this new option falls into the overhead category. As Figure 8.7 illustrates, in RPL-RP, we see a slight increase in the size of DAO packet. This raise depends on the number of siblings that are being included in the packet (8 bytes for each sibling). The most significant difference lies in the case of P-DAO, PDR and P-DAO-ACK (optional), which are solely defined for DAO projection. In Figure 8.7, packet types are associated with the number of bytes each control packet consists of. Fortunately, none of packet types crosses the threshold of 127 bytes, which is IEEE 802.15.4's MTU and there is no need for packet fragmentation. Another important observation is the shrinkage in the size of downward data packets in the source routing header (8 bytes for each hop removed). RPL uses source routed data packets in non-storing mode and as expected we do not see any distinction for Hop-By-Hop mode of addressing the packets.

Now to evaluate the accumulated overhead, it is worth mentioning that the transmission frequency of the P-DAO is defined by the controller. For frequency of P-DAO packets, it will suffice to send P-DAO only when controller gets informed about a topological change not as frequent as basic primitives like DIO. P-DAO ACK is obviously following the same trend as it is sent to acknowledge reception or malfunctioning routes. In our scenario, the controller initialises the data flows which does not necessarily hold true for all the applications, but PDR is also not so frequent since it is only required when asking for a P-DAO. On the other hand, DIO packets are the most frequent RPL packets and are ruled by the *Trickle* algorithm. Therefore, after initializa-

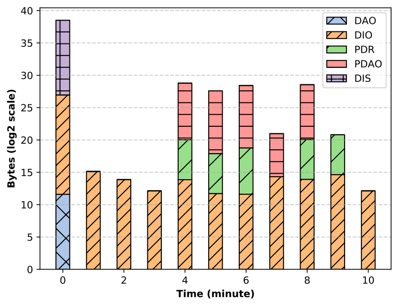


Figure 8.8: Accumulated overhead of control packets in RPL-RP.

Table 8.1: Memory footprint of the routing protocols.

	RPL-RP		RPL	
	ROM	RAM	ROM	RAM
UDP client	44 kB	7340 kB	43 kB	7460 kB
Border Router	170 kB	70 kB	163 kB	70 kB

tion of DODAG, there is still a fair amount of DIO packets in the network. In contrast, DIS and DAO packets are rarely seen after RPL's initial convergence. DIS packets are meant to ask for a DIO from neighbors if the node does not have any route to the root node. DAO packets notify the root of the routing state DAO and DIS are usually exchanged more frequently in the early minutes. To better visualize the results, Figure 8.8 presents a logarithmic scale of control traffic accumulated per minute. DIO packets are the most dominating element followed by DAO and DIS packets. P-DAO and PDR packets appear only after four minutes from the beginning of simulation when the transversal flows start. At this point, according to the logarithmic scale DIO packets generated almost 4 orders of magnitude higher amount of traffic compared to P-DAO and PDR. **Collectively, P-DAO and PDR packets sum up to a 3% of the total control traffic in RPL-RP**.

Furthermore, due to the memory constraints common to low-power devices, it is important to keep track of the memory footprint of RPL-RP. As Table 8.1 shows, there is only about 1 KB increase both in volatile and non-volatile memory of the in-network nodes. The border router can handle more overhead since it is usually deployed on capable devices. Overall, the memory footprint of supporting route projection is inconsequential.

8.5 Conclusion

We presented RPL-RP, an extension to RPL that supports injecting point-to-point routes on-demand by a centralised entity. The new system defines new control packet types and options that collect extra sibling information to be visualized in a dashboard. This enables the administrator to define routing states in the in-network nodes. Overall, evaluation of RPL-RP showed the improvements incurred by the projected routes can surpass its overheads. Although its performance highly depends on the quality of the projected routes, we showed that with a reasonable overhead in control traffic and memory, RPL-RP achieves an almost perfect packet delivery for transversal routes with routes that optimized the latency for hundreds of milliseconds.

Similar to most of the solutions in the related work, RPL-RP only supports installing routes with highest priority and single address destinations (not a range of addresses) which satisfies the requirements of most IoT networks. For the future work, it is worth considering scenarios in which it is necessary or at least useful to install not only high priority routes but also backup routes for fast fail-over as it is supported by OpenFlow.

Bibliography

[1] Tim Winter, Pascal Thubert, Anders Brandt, Jonathan W Hui, Richard Kelsey, Philip Levis, Kris Pister, Rene Struik, Jean-Philippe Vasseur, Roger K Alexander, et al. RPL: IPv6 Routing Protocol for Low-Power and Lossy Networks. *rfc*, 6550:1–157, 2012.

[2] Simon Duquennoy, Joakim Eriksson, and Thiemo Voigt. Five-nines reliable downward routing in rpl. *arXiv preprint arXiv:1710.02324*, 2017.

[3] Hossein Fotouhi, Daniel Moreira, and Mário Alves. mRPL: Boosting mobility in the Internet of Things. *Ad Hoc Networks*, 26:17–35, 2015.

[4] Hossein Fotouhi, Daniel Moreira, Mário Alves, and Patrick Yomsi. mRPL+: A mobility management framework in RPL/6LoWPAN. *Computer Communications*, 104:34–54, 2017.

[5] Bardia Safaei, Aliasghar Mohammadsalehi, Kimia Talaei Khoosani, Saba Zarbaf, Amir Mahdi Hosseini Monazzah, Farzad Samie, Lars Bauer, Jörg Henkel, and Alireza Ejlali. Impacts of mobility models on rpl-based mobile iot infrastructures: An evaluative comparison and survey. *IEEE Access*, 8:167779–167829, 2020.

[6] Philip Levis, Thomas Clausen, Jonathan Hui, Omprakash Gnawali, and JeongGil Ko. The trickle algorithm. *Internet Engineering Task Force, RFC6206*, 2011.

[7] P. Thubert, R.A. Jadhav, and M. Gillmore. Root initiated routing state in RPL. RFC-draft 1654, RFC Editor, July 2020.

[8] Tie Luo, Hwee-Pink Tan, and Tony QS Quek. Sensor openflow: Enabling software-defined wireless sensor networks. *IEEE Communications letters*, 16(11):1896–1899, 2012.

[9] Laura Galluccio, Sebastiano Milardo, Giacomo Morabito, and Sergio Palazzo. SDN-WISE: Design, prototyping and experimentation of a stateful sdn solution for WIreless SEnsor networks. In *2015 IEEE Conference on Computer Communications (INFOCOM)*, pages 513–521. IEEE, 2015.

[10] Pankaj Berde, Matteo Gerola, Jonathan Hart, Yuta Higuchi, Masayoshi Kobayashi, Toshio Koide, Bob Lantz, Brian O'Connor, Pavlin Radoslavov, William Snow, et al. Onos: towards an open, distributed sdn os. In *Proceedings of the third workshop on Hot topics in software defined networking*, pages 1–6, 2014.

[11] Lucia Lo Bello, Alfio Lombardo, Sebastiano Milardo, Gaetano Patti, and Marco Reno. Experimental assessments and analysis of an sdn framework to integrate mobility management in industrial wireless sensor networks. *IEEE Transactions on Industrial Informatics*, 16(8):5586–5595, 2020.

[12] Michael Baddeley, Reza Nejabati, George Oikonomou, Mahesh Sooriyabandara, and Dimitra Simeonidou. Evolving SDN for low-power iot networks. In *2018 4th IEEE Conference on Network Softwarization and Workshops (NetSoft)*, pages 71–79. IEEE, 2018.

[13] Stephen Dawson-Haggerty, Arsalan Tavakoli, and David Culler. Hydro: A hybrid routing protocol for low-power and lossy networks. In *2010 First IEEE International Conference on Smart Grid Communications*, pages 268–273. IEEE, 2010.

[14] Simon Duquennoy, Olaf Landsiedel, and Thiemo Voigt. Let the tree bloom: Scalable opportunistic routing with oRPL. In *Proceedings of the 11th ACM Conference on Embedded Networked Sensor Systems*, pages 1–14, 2013.

[15] Emmanuel Baccelli, Matthias Philipp, and Mukul Goyal. The P2P-RPL routing protocol for IPv6 sensor networks: Testbed experiments. In *SoftCOM 2011, 19th International Conference on Software, Telecommunications and Computer Networks*, pages 1–6. IEEE, 2011.

[16] Mukul Goyal, Emmanuel Baccelli, Matthias Philipp, Anders Brandt, and Jerry Martocci. Reactive discovery of point-to-point routes in low power and lossy networks. *IETF Request For Comments RFC*, 6997, 2013.

[17] Hossein Fotouhi, Maryam Vahabi, Iliar Rabet, Mats Björkman, and Mário Alves. MobiFog: Mobility Management Framework for Fog-assisted IoT Networks. In *IEEE Global Conference on Internet of Things GCIoT'19, 04 Dec 2019, Dubai, United Arab Emirates*, 2019.

[18] Iliar Rabet, Shunmuga Priyan Selvaraju, Hossein Fotouhi, Maryam Vahabi, and Mats Björkman. Poster: Particle filter for handoff prediction in sdn-based iot networks. In *EWSN*, pages 172–173, 2020.

[19] Pascal Thubert, Maria Rita Palattella, and Thomas Engel. 6TiSCH centralized scheduling: When SDN meet IoT. In *2015 IEEE conference on standards for communications and networking (CSCN)*, pages 42–47. IEEE, 2015.

Chapter 9

Paper E:
SDMob: SDN-based Mobility Management for Fog-assisted IoT Networks

Iliar Rabet, Shunmuga Priyan Selvaraju, Hossein Fotouhi, Mário Alves, Maryam Vahabi, Ali Balador, Mats Björkman
MDPI Journal of Sensor and Actuator Networks

Abstract
Internet-of-Things (IoT) applications are envisaged to evolve to support mobility of devices while providing quality of service in the system. To keep the connectivity of the constrained nodes upon topological changes, it is of vital importance to enhance the standard protocol stack, including the Routing Protocol for Lossy Low-power Networks (RPL), with accurate and real-time control decisions. We argue that devising a centralized mobility management solution based on a lightweight Software Defined Networking (SDN) controller provides seamless handoff with reasonable communication overhead. A centralized controller can exploit its global view of the network, computation capacity, and flexibility, to predict and significantly improve the responsiveness of the network. This approach requires the controller to be fed with the required input and to get involved in the distributed operation of the standard RPL. We present SDMob, which is a lightweight SDN-based mobility management architecture that integrates an external controller within a constrained IoT network. SDMob lifts the burden of computation-intensive filtering algorithms away from the resource-constrained nodes to achieve seamless handoffs upon nodes' mobility. The current work extends our previous work, by supporting multiple mobile nodes, networks with a high density of anchors, and varying hop-distance from the controller, as well as harsh and realistic mobility patterns. Through analytical modeling and simulations, we show that SDMob outperforms the baseline RPL and the state-of-the-art ARMOR in terms of packet delivery ratio and end-to-end delay, with an adjustable and tolerable overhead. With SDMob, the network provides close to 100% packet delivery ratio (PDR) for a limited number of mobile nodes, and maintains sub-meter accuracy in localization under random mobility patterns and varying network topologies.

9.1 Introduction

With the advent of the Internet-of-Things (IoT) and its revolutionary role in numerous application domains such as healthcare, industrial automation, and environmental monitoring, there is an increasing demand for seamless support of mobile nodes (MNs). However, the de facto protocols for low-power and lossy networks (LLNs), including Routing Protocol for Lossy Low-power Networks (RPL) and IPv6 over Low -Power Wireless Personal Area Networks (6LoWPAN), have not been designed to cope with highly dynamic network topologies. These protocols are not able to handle rapid topological changes in the network in a timely and accurate manner. Further, it has been shown that various mobility patterns impact the performance of the standard RPL protocol significantly [1]. This occurs due to the continuous relocation of mobile nodes and the delayed readjustments of RPL by the 'trickle' algorithm. In practice, the trickle algorithm is responsible for adapting the transmission frequency of control packets to the rate of the topological changes [2]. Increasing the control packet rate could result in better responsiveness to mobility, but at the cost of higher resource consumption in terms of communication and energy overheads. Nevertheless, since no predictive measure is taken, the mobile nodes' routes are only updated after a period of disconnection, leading to network inaccessibility periods that will cause packets loss and higher delays.

A proactive approach to support seamless hand-off in MNs could rely on Bayesian filters (such as Kalman filters) or other predictive techniques to forecast the future position of MNs [3]. Here, a filter is referred to as the methods that estimate the state of a temporal variable, which is usually observed under noisy measurements [4]. It is common to have a fixed infrastructure of static nodes (in fixed and a priori known positions) that estimate the distance from the MNs. MNs are also usually assisted by an *Inertial Measurement Unit* (IMU) to measure velocity and direction of movement. In such a scenario, it is practical to exploit Bayesian filters to fuse these two sources of information and, thereby, benefit from an accurate localization which leads to improvement in network responsiveness [5].

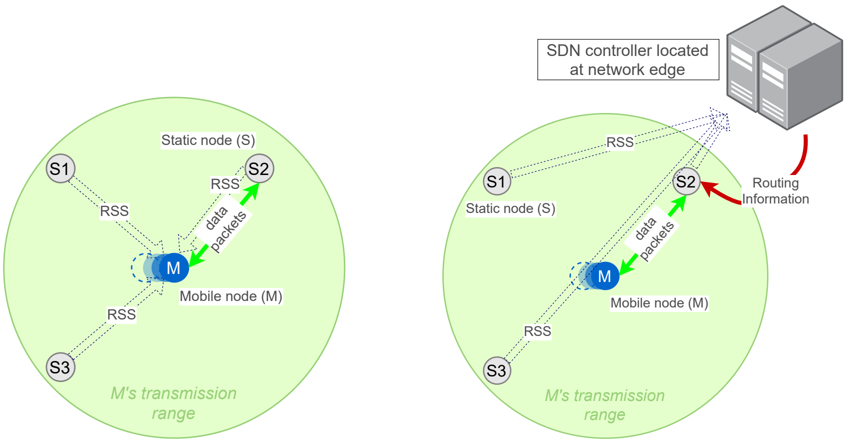
The Kalman filter has been proved to be unbiased (the average error across all the recursive runs is zero), consistent (the filter is neither overconfident nor under-confident) and optimal (it minimizes the estimation error) [4]. However, when using a Kalman filter, the posterior distribution (after the observations) can be computed in closed form only when the relationship between states and observations is described by a linear function, and the measurement and prediction noise follow a Gaussian distribution [6]. To address nonlinear system models, *Extended Kalman Filter* (EKF) was introduced which uses *Taylor se-*

ries to linearize the equations, trading for a negligible approximation error. On the other hand, Unscented Kalman Filter (UKF) and Particle filter have shown higher accuracy in prediction with the bi-modal distribution of the noise [4]. In the literature, these filters are integrated with low power routing protocols mostly in distributed environments [7–9].

The overall architecture of distributed routing algorithms with location prediction in the literature is depicted in Figure 9.1a, where the anchors report back to the MN so it can predict its future parent on its own and modify its routing state accordingly. In this scenario, the accuracy of the location prediction significantly impacts the connectivity of the MN, and to implement accurate and predictive models, we require higher computation capacity than what mainstream IoT devices can afford. In practice, the resource-constrained devices barely manage to support simple data filters, such as Kalman filters. The next important limitation is flexibility in configuration, as the position information is required as an input for the localization of MN, and this can be done much easier with the SDN approach compared to a distributed situation where each node should be fed with this information. We argue that these limitations can be alleviated by offloading the computational burden of these data filters to a centralized external entity, such as an edge/fog device or an external Software-Defined Networking (SDN) controller, depicted in Figure 9.1b. These centralized devices have sufficient resources while being accessible for IoT devices with reasonable latency.

In our context, an edge or fog device is defined as a one-hop resource or service provider. It is placed in close proximity to the client devices to access their resources at a low delay and low accessibility cost. In recent years, there has been a roll-out of edge-based services to improve network reliability, services, and reactivity in the rapidly growing field of IoT applications. Specifically, it has been commonly used by external proprietary real-time localization systems (RTLS) such as Ubisene and Sewio [10] to be implemented in a centralized device at the network edge. SDMob can be integrated into such systems and exploit the information gathered through the distributed operation similar to RPL. However, the involvement of an external SDN controller can also raise new challenges such as additional control packets, leading to control overhead and reservation of system resources to robustly handle them [11].

This paper extends our previous work [12], where we had compared SDMob—*Software Defined Mobility management*, powered by either particle filter or unscented Kalman filter, with the baseline RPL and mRPL [13]. Our previous work only partially addressed mobility patterns (linear and circular trajectories), and scalability of the system in different aspects including the number of MNs, anchors, and their topology (density and distance) were not



(a) (b)

Figure 9.1: Illustration of mobility management strategies—(**a**) Distributed strategy—Mobile node (*M*) does the routing function by processing the Received Signal Strength (*RSS*) values from neighbouring Static nodes (*S*); and (**b**) Centralized strategy—SDN controller at network edge picks the next best parent for *M* to relay data packets.

analyzed. In this work, we introduce new mechanisms to handle congestion in dense networks and to support multiple mobile nodes for enabling seamless handoffs in more realistic scenarios. The new mechanisms include smart buffer management to support multiple MNs, new timers for congestion and restructured route-installation packet format. We also evaluate the impact of other system parameters such as path loss variance and the velocity of the MNs. In this work, we have also included the analytical evaluation of the system in comparison with the simulation results.

The main contributions of this work are listed below:

- Enhancement of the SDMob architecture by introducing new timers and route installation format to achieve a better quality of service (QoS) in networks with varying topologies, realistic mobility patterns, and supporting multiple MNs.
- Comparison of SDMob with the benchmark ARMOR and baseline RPL.
- Modelling of the SDMob architecture through probabilistic analysis and comparing the analytical results with the simulation results.
- Evaluation of SDMob in various conditions, considering different mobility patterns, link quality fluctuations as well as network scalability in

terms of hop distance from the MN to the controller, number of neighbors, and number of mobile nodes.

- Implementation of SDMob in the Cooja/Contiki environment, where the code is available online `https://github.com/iliar-rabet/sdmob`, (accessed on 18-01-2022).

The rest of the paper is organized as follows: Section 9.2 provides a brief description of RPL limitations upon mobility, and outlines some efforts to overcome them. Section 9.3 describes the SDMob architecture, and the employed filter for data processing. Section 9.4 takes an analytical approach to evaluate the system and measure the probability of packet loss during handoff. Section 9.5 describes the simulation environment and performance evaluation, comparing SDMob with the selected benchmark (ARMOR). Finally, in Section 9.6 conclusions are drawn.

9.2 Related Works

This section provides an overview of some of the main related works, which are specifically focused on mobility support in RPL, localization algorithms within the routing protocol, and edge or fog computing architecture in IoT networks enabled with SDN devices.

9.2.1 Overview of RPL.

RPL is considered as the de facto routing protocol for IoT. RPL maintains a distributed data structure, named Destination Oriented Directed Acyclic Graph (DODAG). The process starts with the root node transmitting a DODAG Information Object (DIO) that embeds the necessary information to construct a route towards the root (rank 0 for root node). Upon receiving the DIO packet, each node selects its preferred parent (based on some objective function) and schedules relaying a DIO packet with its non-decreasing rank to further advertise the network. Upward traffic can be routed after DIO packet transmission, but for downward routing, the root node (or parent in storing mode) gets notified about the high-rank nodes only after transmission of Destination Advertisement Object (DAO) packets.

RPL allows two modes of operation— storing and non-storing—for downstream traffic. In non-storing mode, it is only the root that maintains the downward routes. This mode scales better, since the memory footprint at intermediate nodes does not increase with the size of the network. In RPL, it is more challenging to support mobility for downstream traffic since a mobile node must notify the root (rather than only updating its parent for upstream traffic). SDMob uses the non-storing mode of RPL so that the controller can manipulate the source-routed downward packets.

In the original RPL, the most common Objective Functions (OFs) are (1) the Minimum Rank Hysteresis Objective Function (MRHOF), and (2) Objective Function Zero (OF0). Objective functions define how RPL nodes minimize the given routing metric: hop count, expected transmission count (ETX), or latency. MRHOF is designed to minimize the path cost while avoiding excessive churns in the network. The nodes using MRHOF only switch the best parent if the minimum is improved by a threshold (this is also known as the hysteresis mechanism) [14]. OF0 is faster in terms of updating the routes in a dynamic environment as it changes the best parent even after slight improvements in the routing metric [15]. Our proposed mobility solution (SDMob) does not enforce any specific requirement on the objective function or the routing metric. Rather, SDMob allows the baseline version of RPL to converge with arbitrary metrics and then injects higher priority routes from the controller.

9.2.2 Mobility-aware RPL routing.

Mobility creates an inconvenience for all layers of the protocol stack, especially the routing protocol. While RPL supports infrequent joining and leaving of nodes, it performs poorly upon the dynamics imposed on the network topology. The authors in [16], survey the enhancements made to RPL to support mobility and focus on the specific mechanisms that have been altered. A more recent survey [17] classifies mobility extensions of RPL into solutions including (i) only mobile nodes (e.g., VANETs) and (ii) those with a fixed infrastructure as well as mobile nodes.

One of the extensions of RPL that addresses mobility of nodes is BRPL [18], which combines backpressure routing [19] with the objective functions of RPL. To support high-throughput traffic, BRPL takes into consideration the queue backlogs of the neighbors. This allows BRPL to utilize sub-optimal routes when the optimal route is congested.

In mRPL [13], MN operates in two phases of data transmission and discovery. In the *Data Transmission phase*, static nodes (SNs) constantly monitor the link quality and compare the Received Signal Strength Indication (RSSI) measurements to a threshold T_l that indicates the minimum RSSI threshold in a reliable channel. If the link quality crosses the threshold, the SNs notify the MN with a beacon, and then MN stops transmitting data packets and instead starts the *Discovery Phase* by sending DIS packets to request the neighboring

SNs to respond with DIO packets. Then the MN analyses the received DIO packets, and if the RSSI values from other SNs are higher than T_h, it performs the handoff process and resumes data transmission, otherwise, it continues sending DIS packets. An enhanced version called mRPL+ [20], relies on overhearing of mobile nodes' packets by alternative parents. Then mRPL+ takes a similar approach to mRPL by selecting the preferred parent based on some thresholds.

For the case where all nodes are mobile, Tian et al. [21] adjust the Trickle timer according to mobile nodes' velocity, and utilize geographical information as the routing metric. If the mobile nodes are not equipped with IMU sensors, *Doppler Effect* can be used to estimate their velocity as explained in [22]. Murali et al. [23] introduce D-trickle to support mobility, where the chosen DIO interval depends on the number of neighboring nodes. In a dense network, D-trickle sends DIO packets less frequently but for a sparse network, frequent DIO packets boost the connectivity of the network.

Ancillotti et al. [24] propose RL-Probe for using reinforcement learning methods to determine when to send probing packets to estimate link qualities. Using RL and multi-armed bandit theory in RPL minimizes the communication overhead while keeping the network responsiveness at a high level. Another approach has been implemented in GTM-RPL, which is based on game theory to select the optimal transmission rate of the nodes [25]. The authors prove the existence of Nash Equilibrium. In other words, each node can reach an optimal strategy with no incentive to change while other nodes keep their current strategy. In the next step, nodes select the preferred transmission rate based on the mobility of nodes (detected by RSSI) and other parameters. This approach is only practical in applications where the transmission rates can be manipulated.

ARMOR [26] is a recent work that calculates the time each parent is available based on location data and selects the node with the so-called longest "Time-To-Reside". We use ARMOR as a benchmark in our work, and we will further explain it later in Section 9.5.1. RMA-RP [27] introduces a similar metric called "Time-To-Stay" using only two recent RSSI measurements. The choice of this metric is debatable since these two models may choose links that are active for a long time but are lossy since stability is interpreted with the duration of the connection rather than the link quality. This metric manages to reduce the number of performed handoffs and preserves energy consumption. RMA-RP also modifies the DIO intervals to be lower for low-rank nodes (close to the root) as they provide connectivity for the other nodes.

9.2.3 Location estimation models for enhancing routing protocols.

A class of extensions of RPL consists in boosting the mobility support by integrating a filtering/localization method into the routing protocol. For instance, in [7], authors have proposed Kalman-RPL to predict the future position of the mobile node and consequently estimate the future link qualities. In Kalman-RPL a mobile node transmits a beacon that includes its velocity information in specific intervals and is responded to by the receiving static nodes. After a positioning phase that estimates the current position of the mobile node using three static nodes in its vicinity, MN can predict its future position. EKF-RPL [8] takes a similar approach to Kalman-RPL, but it employs Extended Kalman Filter (EKF) within RPL to better support non-linear trajectories.

Some research works apply the Kalman filter for predicting link quality regardless of the routing protocol. Parasuraman et al. [28] model path loss and shadow fading of the wireless channel independently and then apply Kalman filter to fuse both models.

There are also some efforts on adopting *on-demand* routing strategies when a node starts searching for a route for transmitting data. The *Lightweight On-Demand Adhoc Distance-vector* routing protocol - Next Generation (LOADng) [29] is one such protocol specifically designed to support any-to-any communication in LLNs, which has received less attention compared to RPL. EKF-LOADng [9] enhances basic LOADng by predicting the link's RSSI after a positioning phase (triangulation algorithm) and running the EKF. In the triangulation phase, a mobile node broadcasts a message asking for packets from its static neighbors. Responses from static nodes experience a random waiting time to avoid collisions.

Compared to the EKF, particle filter leads to more accurate results and better resiliency against nonlinear moving trajectory and non-Gaussian noise [4]. The particle filter, which is also known as *Sequential Monte Carlo* uses hundreds to thousands of samples to predict the future state and fuse the measurement, hence requiring a higher computation capacity than most constrained IoT nodes can provide. In the context of cellular communications, the particle filter has been used in the literature to spot the mobile nodes and select the best service point as in [30]. This work also proposes a Rao-Blackwellised particle filter as a lightweight alternative to the baseline particle filter.

In a previous work [12], we proposed and implemented the basic version of SDMob for offloading the mobility solution to the edge devices in a centralized manner. The basic SDMob outperformed RPL and mRPL [13] in a topology with a single MN. This paper presents enhanced SDMob, supporting networks with multiple mobile nodes, high density of anchors, varying hop distances, and realistic mobility patterns.

9.2.4 SDN-enabled IoT network architectures.

There is growing popularity in using SDN-enabled solutions in IoT networks and a recent survey [31] addresses the proposals implemented by the community to integrate the SDN in IoT networking in a coherent and lossless manner. It is too expensive to simply integrate the common SDN solutions and standards within constrained IoT networks without re-designing the SDN to consider IoT limitations [32]. Therefore, there is a requirement for devising solutions targeting IoT networks with reduced complexity and operational cost.

Efforts have been made (including by standardization bodies) to design solutions for managing IoT networks. The *Internet Engineering Task Force* (IETF) has a recent draft for infusing data routes into the network that is called DAO projection [33]. It defines a framework for the root node to initialize some options in DODAG Advertisement Objects (DAO) through new control messages, namely Project DAO Request (PDR) and PDR-Acknowledgement (PDR-ACK). This enables the root node to install routes in either the source or intermediate nodes along the path. The mechanism is a low-overhead substitute for implementing centralized network management in IoT networks. We have presented a similar implementation of this draft in RPL-RP [34] that aims at optimizing any-to-any routes.

Coral SDN [35] is another RPL-based solution that allows an SDN controller to manipulate RPL routing parameters such as the interval used by the Trickle timer. The interval is the duration between successive DIS messages from a leaf node, which is an important configuration to adapt the responsiveness of the network.

Theodorou et al. [36] proposed SD-MIoT, in which RPL is assisted by an SDN controller. The SDN controller is responsible to detect the mobility of the nodes by maintaining an adjacency graph. The mobility detector assumes the node to be mobile if more than one row in the adjacency graph changes compared to the previous time step, but with a single connectivity change, it can not be determined if the node is mobile. Next, k-means clustering is performed to separate mobile and static nodes. The SDN will then constantly use the adjacency graph to update the forwarding rules.

In μSDN [37], the authors argue that an appropriate design for the centralized controller is to rely on the legacy RPL for basic connectivity. To deal with constraints in IoT networks, it introduced optimizations including avoiding packet fragmentation, source-routed control packets, and timers for fine-tuning.

SDN-WISE [38] is one of the research works on SDN-based low-power networks that has attracted some attention. It installs finite state machines on the constrained nodes to handle the rules. For connecting the controller to the mesh network it adds a layer called Topology Discovery that is responsible to interpret packets from neighbors that advertise their hop distance to the sink and remaining battery power. In some other works, such as μSDN, SDMob, and DAO projection, this basic connection is prepared by the RPL protocol. Some extensions to SDN-WISE try to address mobility by assigning a MAC layer schedule [39] or by using multicast routing, similar to MMF-SDN [40].

Mertens et al. [41] proposed SDN-(UAV)ISE in which a drone acts as a mobile sink (a.k.a mule). The SDN controller, which is based on SDN-WISE applies a decision tree learning algorithm on the data from sensors to predict the position of the drone. They also optimize the destinations that the drone is supposed to visit and this optimization can be reduced to the NP-complete "set cover problem".

MobiFog [42] is our previous work on centralized mobility management, where the discovery of alternative parents is performed using the RSSI measured from data packets. The predictions are independent of the position of the nodes and hence MobiFog does not require prior knowledge about the position of the static nodes. Generally, the approaches that rely on the data traffic including MobiFog and mRPL impose much less overhead, but the handoff process will depend on the traffic at the time of the handoff.

We have summarized the presented works in Table 9.1. Note that when it is mentioned in the table that all of the nodes can be mobile, it means that the protocol design does not require a set of static nodes, yet it does not mean that the authors have claimed or tested the work in such a scenario. Overall, the literature in IoT networks mostly neglects more sophisticated and computation-intensive filters for mobility management. Employing the SDN architecture to overcome challenges raised by the mobility of nodes remains a research gap that we address by introducing SDMob. We believe that SDMob paves the ground for employing more accurate filter/localization algorithms at the SDN controller towards improved performance upon mobility in IoT networks.

9.3 SDMob Architecture

In this section, the structure of the SDMob architecture is addressed in more detail. First, we review the basic design of SDMob, which was presented in the previous work [12]. Next, we describe the mechanisms introduced in the

Table 9.1: Mobility management extensions for RPL.

Mobility Solutions	**Infrastructure**	**Metric**	**Predictive Mechanism**
mRPL [13]	fixed and mobile	ETX and RSSI	average RSSI/SNR
mRPL+ [20]	fixed and mobile	ETX and RSSI	average RSSI/SNR (overhearing)
ARMOR [26]	all can be mobile	Time To Reside	Relative velocity
RMA-RP [27]	fixed and mobile	Time To Stay	2 consecutive RSSI values
D-trickle [23]	all can be mobile	ETX,ELT,RSSI,distance	-
Kalman-RPL [7]	fixed and mobile	predicted ETX	Kalman Filter
EKF-RPL [8]	fixed and mobile	position of MN	Extended Kalman Filter
EKF-LOADng [9]	fixed and mobile	position of MN	Extended Kalman Filter
DAO projection [33]	No mobility	priority for projected routes	-
Coral SDN [35]	fixed and mobile	OF and trickle set by controller	-
SD-MIoT [36]	fixed and mobile	link quality	proactive route installation
SDN-UAise [41]	Mobile sink	not RPL-based	Decision tree
MobiFog [42]	fixed and mobile	ETX and RSSI	Average RSSI
MMF-SDN [40]	fixed and mobile	not RPL-based	-
FTS-SDN [39]	fixed and mobile	not RPL-based	-
BRPL [18]	all can be mobile	backlog drift plus ETX	Lyapunov Optimization
GTM-RPL [25]	fixed and mobile	ETX	Nash Equilibrium
RL-Probe [24]	all can be mobile	ETX (same as RPL)	epsilon-greedy learning

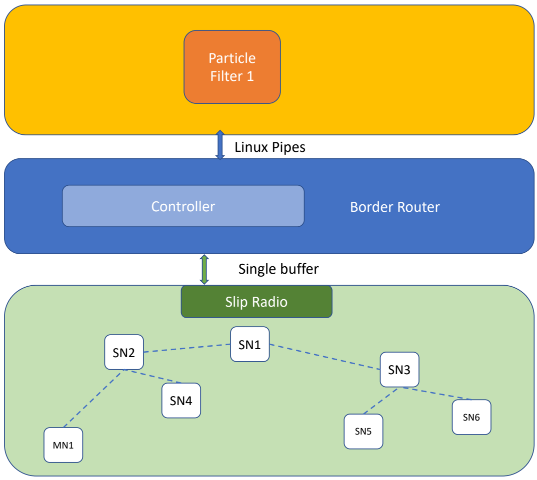
enhanced SDMob to address the challenges that arise in networks with multiple mobile nodes, high density, and more realistic mobility patterns. For simplicity, we are using SDMob for the enhanced version throughout this paper. Finally, we analyze the tracking algorithm employed in the paper (particle filter).

9.3.1 Basic SDMob Architecture

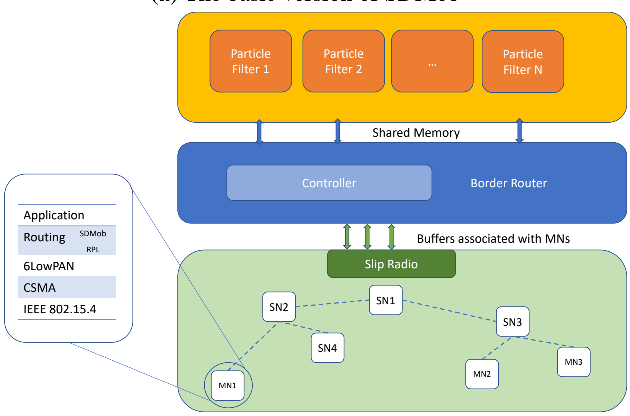
In this subsection, we review the basic SDMob architecture that was presented in a previous work [12]. Figure 9.2 illustrates and compares the basic SDMob against the enhanced version. Inside the WSN network, SDMob uses Contiki's protocol stack including RPL/6LoWPAN networking with a 64-bit network prefix and IEEE 802.15.4. The WSN is composed of one MN in the basic SDMob (multiple MNs in the enhanced SDMob), and Static Nodes (SN). SNs are also known as anchors and provide multi-hop access for the MNs, and provide the controller with the required information for localization. The SDN controller is placed outside of the constrained network, separately as a Linux-server process. It admits to the control information collected via the sink node in the WSN simulated network. The SDN controller is bundled with the border router, which is responsible to decode messages from the low-power domain to be used by the controller and data server. The border router is also a Linux process that is connected to the WSN with the help of a node called Serial Line Internet Protocol (SLIP) radio. The Contiki's implementation of the SLIP protocol prepares the wireless IP packets to be transmitted over a serial line. The particle filter runs the location estimation algorithm to accurately locate future positions of the mobile node. The implementation of the border router and the particle filter are interfaced using Linux pipes (shared memory in the enhanced SDMob due to its fast inter-process communication).

In this paper, we assume the MNs are equipped with Inertial Measurement Unit (IMU) sensors, but the architecture can be generalized to nodes that do not benefit from IMU sensors by estimating the velocity based on previous RSSI measurements. However, measuring the velocity is more accurate compared to the estimation techniques, since it exploits the extra information from the sensors. On the other hand, many applications nodes are already using such sensors, so the overhead could be negligible.

This design is highly inspired by the ideas in an IETF draft titled "root-initiated routing state" [33] that allows a centralized entity to manipulate the routing states in the distributed operation of RPL. The routing rules that are installed by the controller are of a higher priority compared to the routes learned by receiving DIO packets. The SDN controller's traffic relies on RPL for its ba-



(a) The basic version of SDMob



(b) Enhanced SDMob

Figure 9.2: A schematic view of the basic SDMob architecture in (**a**) supporting a single mobile node compared with enhanced SDMob (**b**) with multiple mobile nodes and buffers.

sic connectivity. It is only after the convergence of the standard RPL protocol that the MN can utilize the upward/downward routes to access the controller and the centralized tracking algorithm can notify the anchors to relay MN's traffic.

For the smooth operation of SDMob, certain mechanisms are required. Below we describe some of the challenges and the mechanisms adopted in basic SDMob to address them.

Collision avoidance between control plane and data plane.

The SDN-based architecture comes with an inevitable control overhead that can saturate the MAC layer congestion mechanism in both upward and downward directions. To streamline the traffic, we have implemented a reserved period for control packets called Control Window (CW) out of which transmission of control packets is prohibited. CW can be adjusted based on network conditions; for instance in a network with a long delay between the MN and the controller, it is advised to select a longer CW to allow the control packets to reach the controller. A long CW limits the available time for data packets, and hence data packets are expected to be further delayed.

RPL-Aware Leaf (RAL).

In SDMob installing a new rule on the MNs is troublesome since their links are lossy. A recent IETF draft [43] defines RALs as a host that does not participate in further advertising the DODAG and relies on the RPL routers to forward its traffic. This can solve the problem of routing loops that could happen in the network, for example when MN is denoted as the parent node of another node. Another upside is the reduced memory footprint in the MN. Most importantly, it is beneficial as the controller is mandated to send routes to more reliable SNs rather than using MN's lossy links.

Downward routing process.

Standard RPL favors upstream traffic as it is the predominant traffic pattern in the IoT domain. Most enhancements made to RPL also have weaker behavior when it comes to point-to-point or downward traffic. These shortcomings stem from design choices in RPL. If there is a topological change deep in the network, the root node will only be notified after a timer for sending the DAO packet expires. SDMob relies on the downward routes provided by RPL to relay the controller's commands. It can also reinforce the downward routes by the RSSI measurements that it collects.

Integration to the objective functions.

Once the controller runs the filter and announces the new best parent, it will start serving the MN in the data window. Selecting the closest parent to the MN can be a naive strategy since the characteristics of the multi-hop path will be ignored. An alternative approach is to estimate the one-hop link quality (reverse relation with distance) and use it by adding to the classic metrics such as ETX.

9.3.2 Enhanced SDMob

Previously, basic SDMob design has focused merely on offloading the mobility solution to the resource-rich controller, yet it fails at performing the handoff in some challenging conditions. In this paper, we aim at improving and evaluating the scalability of the proposed system. We review those challenging conditions and introduce mechanisms to overcome those challenges. Table 9.2 summarizes these features and associates them with the versions of SDMob.

Handling anchor density.

The RSSI measurements that provide the vital information for localization generate bursty traffic since all the anchors try to forward the control beacon as soon as they receive it. The constrained network cannot always deliver this bursty traffic to the controller especially if MN resides in a dense area. Although the CSMA-based MAC layer handles the shared spectrum but with bursty traffic in a dense network the delivery ratio drops significantly. With more burstiness in the traffic, CSMA is forced to retransmit more packets due to congestion in the network. This bursty traffic can push the CSMA-based MAC layer to its limits. If the number of retransmissions for a certain packet crosses a certain threshold, the packet will be dropped. Some mobility patterns tend to keep the mobile node in areas with more density. The situation gets worse with multiple mobile nodes moving in the same area.

The accuracy of the filter substantially depends on the RSSI reports arriving on time. However, this bursty traffic can be lost in the anchors or be delayed. By enabling the re-sampling procedure, the particle filter can converge again after a period of not receiving any/enough measurements. But most importantly, we introduce a timer called the congestion timer. The congestion timer randomly delays the transmission of the RSSI report to the controller in order to decrease the burstiness. This timer can be tuned based on the density of the neighbors but excessively increasing this timer may increase the handover delay. By setting the congestion timer to a reasonable value, the delivery ratio at the control plane gets boosted but on the other hand setting it too high may lead to not receiving the packets in time for the filter.

Handling multiple MNs (smart buffer management).

One of the most important features of enhanced SDMob is supporting multiple MNs that require multiple instances of the filter to run in parallel. On the other hand, with multiple MNs the control traffic increases dramatically sometimes even more than the capacity of the network. SDMob will require multiple independent buffers at the border router each associated with an individual MN. These buffers assign a priority to the newest reports and discard the old ones.

Another additional feature focuses on improving the efficiency of the inter-process communication between the border router and the filter. In the enhanced SDMob, the buffers (between the border router and the particle filter) are implemented using shared memory, instead of using Linux pipe files that were utilized in the basic SDMob. This feature drastically decreases the time spent for predicting the future parent and handoff delay, thus improving network reliability.

To increase the reception ratio of the control traffic, we implement a buffer timer that specifies the time that the border router waits for RSSI measurements. This timer can be tuned in accordance with the hop-distance of the mobile node, congestion in the network, congestion timer, and the interval for sending the control beacons. Each MN will have a buffer timer that can be tuned independently.

New route projection packet format.

The packet format of the controller has some impact on the handoff process, mainly through notifying the previous parents. It is important to stop the previous parent from serving the MN immediately after switching to a new parent otherwise two parents will forward the data packets. This may be useful if the number of redundant parents does not increase in a way that too many replicas of the same packet congest the network. In some of the related works, uninstalling root-initiated routes is performed using timers, meaning that the routes are active for a specific time set by the controller. In our previous work, the controller sent the IP address of the MN's preferred parent to all potential parents. Taking these two approaches limits the flexibility in network management, but with low additional overhead due to the SET/UNSET commands in the current work, parents are managed independently.

Table 9.2: Comparing main features of basic SDMob and enhanced SDMob.

Feature	Basic SDMob	Enhanced SDMob
Control Window	✓	✓
RPL Aware Leaf	✓	✓
Downward routing	✓	✓
Integration with OF	✓	✓
Multi-instance filter	×	✓
Multiple buffers	×	✓
Buffer Timer	×	✓
Congestion Timer	×	✓
SET/UNSET commands	×	✓

In Figure 9.3, a timeline demonstration of the SDMob's handoff process is illustrated. The handoff process is carried out in 4 steps, as described below:

- Step 1: MN embeds its velocity in a control beacon and broadcasts it (during the Control Window).
- Step 2: All SNs in the vicinity of MN receive the beacons, append the measured RSSI values and transmit the beacon to the border router. Contention for the wireless channel towards the root node may overwhelm the network. To deal with this problem, we define a random congestion timer that randomly delays forwarding the RSSI values between 5 to 20 milliseconds. The SNs only relay the control beacon after the congestion timer expires.
- Step 3: Another challenge is deciding the amount of time that the border router waits for the RSSI values. We define a second timer called the buffer timer in a deterministic manner that specifies how long the border router waits for control packets. This timer can be tuned based on the distance from the mobile node to the controller and is set to 500 milliseconds by default. Once the buffer timer expires, the border router sends the accumulated buffer over the serial line to the controller for processing, and meanwhile, it assigns each RSSI measurement to its associated buffer.
- Step 4: Controller executes the particle filter, and selects the new best parent. Next, the new and old best parent get notified by a downstream packet containing a SET and UNSET command respectively. The new parent will continue serving the MN's data packets (during the data window) until it receives an UNSET command.

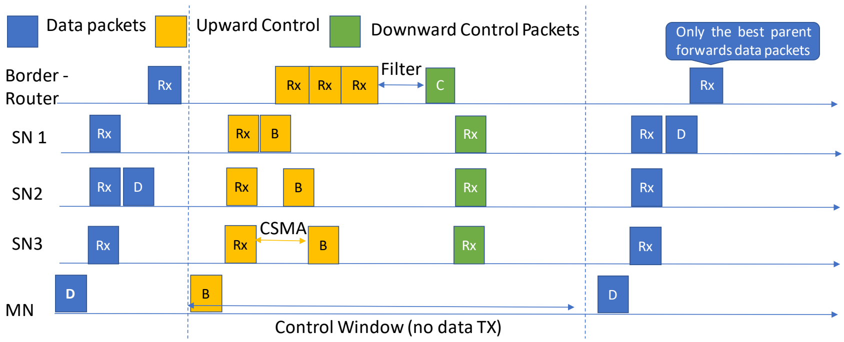


Figure 9.3: Time diagram of handoff process in SDMob showing how buffer timer and congestion timer and SET/UNSET commands improve the basic SDMob [12].

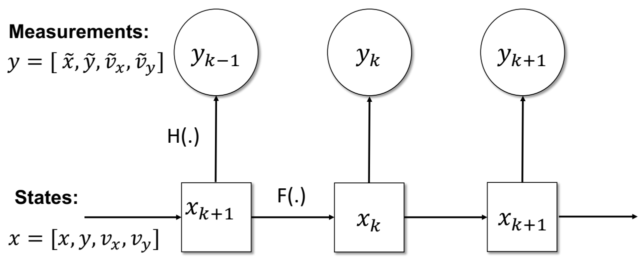


Figure 9.4: Markovian dependencies for the tracking problem.

9.3.3 Filter Design

Tracking algorithms extend the localization of mobile entities in time via successive runs of localization and predictions. Higher error in localization/tracking of the mobile node leads to an erroneous calculation of the link quality and thus sub-optimal selection of parent nodes. The SDMob architecture offloads performing the filter to the more resourceful edge devices.

Bayesian filters such as Kalman and particle filter, model the state space as a Hidden Markov Model as depicted in Figure 9.4. A hidden Markov model is a graphical statistical model that relates the unobservable states (actual position) to the previous states (by velocity) and the noisy observations (based on RSSI reports). Within the model, the Markovian property holds, meaning that each state (k-th) at a given time only depends on the previous state ($k-1$-th).

The state vector and measurement vector at k-th time step are $\mathbf{x}_k = [x \quad y \quad v_x \quad v_y]^T$ and $\mathbf{y}_k = [\tilde{x} \quad \tilde{y} \quad \tilde{v}_x \quad \tilde{v}_y]^T$ respectively. x and y are positions within Cartesian coordinates. The transitional probabilities in a Markov process can be formulated using the Chapman Kolmogorov equation. Given a set of measurements before the current time slot $z_{1:k-1} = z_1, ..., z_{k-1}$, the prior probability or $p(x_k|z_{1:k-1})$ we have:

$$p(x_k|z1:k-1) = \int R^{n_x} p(x_k|x_{k-1})p(x_{k-1}|z1:k-1)dx_{k-1} \quad (9.1)$$

And then after receiving the measurement z_k, the posterior probability $p(x_k|z_{1:k})$ can be given by Bayes rule:

$$p(x_k|z1:k) = \frac{p(z_k|x_k)p(x_k|z_{1:k-1})}{p(z_k|z_{z:k-1})} \quad (9.2)$$

The state vector can be related to the previous state (Markov Property) using a function denoted by F in Equation (9.3) and the relation of each state with the corresponding measurements can be described using a function denoted by H in Equation (9.4).

$$\mathbf{x}_k = F(\mathbf{x}_{k-1}) + \mathbf{n}_{k-1} \quad (9.3)$$

$$\mathbf{y}_k = H(\mathbf{x}_k) + \mathbf{r}_k \quad (9.4)$$

In this system model, $\mathbf{n}_k$ and $\mathbf{r}_k$ are the prediction and measurement noise respectively. These two noises are mutually independent. These equations are only analytically tractable if noise is Gaussian and functions are linear and Kalman Filter is only advantageous in linear functions and the presence of Gaussian noise. Some enhancements such as Enhanced Kalman Filter (EKF) focus on handling non-linear F and H functions. However, to counteract non-Gaussian noise under a non-linear trajectory in the controller, we are compelled to adopt some other techniques such as UKF or particle filter.

Particle filter is also known as Monte Carlo Sampling and maintains a set of fully random particles as in Equation (9.5). The filter can take advantage of any a priori knowledge of the obstacles and infeasible positions when initializing the samples.

$$p(x_t) = \sum_{i=1}^{N} w_i \delta(x - x_i) \quad (9.5)$$

Once it receives the RSSI measurements, first the distance can be estimated based on the radio model. Using the path loss model in Equation (9.7), the controller estimates the distance between MNs and distinct SNs and then applies triangulation. Here, P_1 denotes the received signal strength in a 1 meter distance, and α is a constant value, describing the radio propagation in the environment [44].

$$RSSI_{(d,t)} = RSSI_{d_0} - 10\eta log_{10}(\frac{d}{d_0}) + X_\sigma \quad (9.6)$$

$$d = 10^{\frac{RSSI - RSSI_{d_0}}{10\eta}} \tag{9.7}$$

Then the filter updates the weights based on

$$w_i = \frac{\tilde{w_i}}{\sum_{i=1} N\tilde{w_i}} \tag{9.8}$$

In the prediction step, the filter moves all the particles based on the IMU information. In this step, the particle filter defines obstacles or feasible areas for the MN by simply removing the samples that come to be outside the legit area. Consequently, an estimate of the mean of the posterior is calculated using simple weighted averaging. Finally, the filter selects one of the anchors to become the new preferred parent and initiates a route-projection packet, containing either a SET or UNSET rule. The SET rule is sent to the new parent to suggest accepting the MN's data packets and the UNSET rule signals the anchor to stop relaying. If an UNSET packet gets lost, it may cause the network to carry multiple instances of the same data packet and congest the network. But one can arrange such a multi-parenting scenario deliberately to have redundant anchors for a mobile node. Studying multi-parent routing is out of the context of this paper. SDMob sends the UNSET packet only if the preferred parent has changed, and the SET packet is transmitted every filtering interval.

With the probabilistic approach that particle filter takes, some of the samples may lose their importance and cause the filter to diverge. The filter should eliminate those irrelevant particles, and increase the number of credible particles. This can be performed in the re-sampling step, and to avoid the overheads it is carried out only when the number of effective samples drops below a certain threshold. There are a few alternative algorithms that the filter can select as re-sampling strategies such as systematic, stratified, and residual re-sampling. In SDMob we use systematic re-sampling in which N points are selected (with even distances) in the whole area and then randomly moved. It is believed that systematic re-sampling reduces the computational complexity while giving identical or improved estimates [45]. The number of effective samples can be defined using Equation (9.9).

$$N_{eff} = \frac{1}{\sum_{i=1}^{N} w_k^{(i)}} \tag{9.9}$$

All the mentioned steps are performed for each MN, and each MN has its own set of particles. Each RSSI measurement contains the IP address of the MN, so the filter can easily associate RSSI values to the particle sets. The overall procedure is presented in Algorithm 1.

Algorithm 1 SDMob's algorithm with Particle Filter

Initialize a number of particles with random values for positions and velocities

$$X_0^i \leftarrow p(x_0)$$

while Termination condition not reached **do**
  Wait for the filter timer
  for each Mobile Node **do**
    if $N_{eff} \leq N/2$ **then**
      Re-sample
    end if
    Update weights:
    for Each particle i **do**
      $\mathcal{L}(y_t|x_{t-1}^i) \leftarrow N(H(x_k), \theta)$
      $\tilde{w}_t^i \leftarrow w_{t-1}^i \mathcal{L}(y_t|x_{t-1}^i)$
    end for
    Predict samples:
    for Each particle i **do**
      $x_k^i \leftarrow F(x_{k-1}^i)$
    end for
    Estimate by weighted averaging
    Send SET/UNSET packets to the Mobile Node
  end for
end while

9.4 Analytical Model and Evaluation

We have conducted a probabilistic analysis of the SDMob architecture. This model can guide us through the reasons why SDMob may lose or gain advantages in specific scenarios. In a previous work [46], we focused on comparing RPL with SDMob using the analytical evaluation. Now we aim at evaluating the impact of parameters such as path loss variance and the MN's velocity, and compare the results from the analytical model with simulation results.

9.4.1 Analytical Model

For modeling the radio and network, we try to resemble the simulations that are explained in the next section. The received signal strength at distance d can be estimated using Equation (9.7) with σ being the standard deviation in RSSI measurements due to shadowing. We consider a scenario in which an MN moves from the proximity of static node A, approaching its future parent, static node B that resides in a 10-meter distance, with a constant speed of V horizontally as depicted in Figure 9.5a. The transmission range of the nodes is 5 m. The upward control beacons are assumed to be delivered to the controller, and the particle filter tracks the MN (with positioning error = 0). The downward packet containing the new routing rule from the controller (route-projection packet) is assumed to be received at the MN at time T_rx which follows a normal distribution. The expected signal quality ($\overline{R_A(t)}$) from node A deteriorates exponentially in time with increasing distance and contrarily rises for node B (see Figure 9.5b).

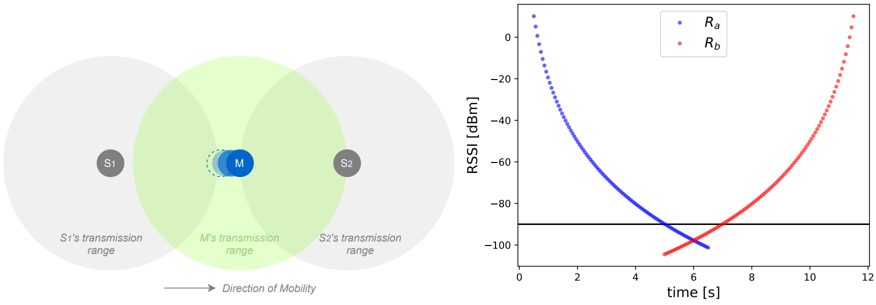
Since the distribution of the noise (X_σ) follows the Gaussian distribution, the probability of the RSSI to be below a certain threshold (T_ℓ) can be derived by:

$$P(R_a(i) < T_\ell) = Q(\frac{-T_\ell + \overline{R_a(i)}}{\sigma})$$

where T_ℓ corresponds to the lower threshold of RSSI and Q function is the complementary distribution function of the standard Gaussian distribution.

$$Q(x) = \frac{1}{\sqrt{2\pi}} \int_x^\infty \frac{exp(u^2)}{2} du \tag{9.10}$$

Accordingly, the probability of packet loss at time t can be estimated. If the route-projection packet is not received at time t ($T_{rx} > t$), probability of packet loss equals ($R_a(t) < T_\ell$) meaning that if the link to the old parent gets disconnected. After reception of the route-projection packet, it is the link to



(a) (b)

Figure 9.5: The illustration of the scenario considered for the analytical evaluation—(**a**) mobile node M moves from vicinity of its current parent (S1) to the future parent (S2); (**b**) expected RSSI values with respect to the lower threshold of RSSI when M starts moving from vicinity of AP_a to AP_b (right).

parent B that matters and packets can get lost only if $R_b(t) < T_\ell$. We assume that the probability of receiving the route projection packet in time follows a normal distribution. Since the reception of the route-projection packet is independent of the link quality, we can derive the packet loss probability by multiplying these two as formulated in Equation (9.11).

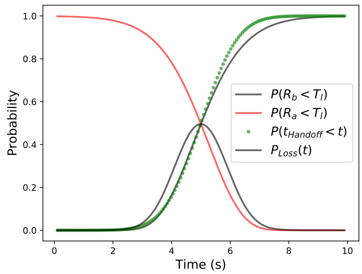
$$\begin{aligned} P_{Loss}(t) = P(R_a(t) < T_\ell) \times P(T_{rx} < t) \\ + P(R_b(t) < T_\ell) \times P(T_{rx} > t)) \end{aligned} \tag{9.11}$$

where the $P(T_{rx} < t))$ is simply the integral of the probability distribution function of T_{rx}:

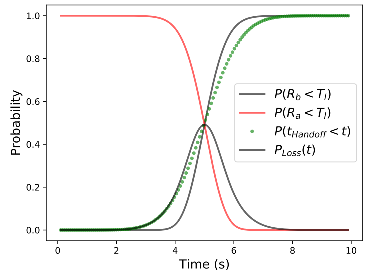
$$P(T_{rx} < t) = \int_0^t P(T_{rx=\theta}) d\theta) \tag{9.12}$$

9.4.2 Analytical Evaluation

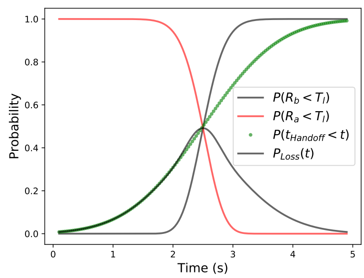
With the model being devised, we can determine the probability of packet loss when changing the model parameters. With higher standard deviation in the RSSI measurements, it is expected that link quality fluctuations are observed more often and the probability of individual links being broken increases. As illustrated in Figure 9.6a,b, $P_{Loss}(t)$ increases with σ since it is the summation of availability of individual links according to Equation (9.11). In Figure 9.6c it can be seen that with increased velocity (2 m/s), $P_{Loss}(t)$ increases compared to the case with 1 m/s and the same σ depicted in Figure 9.6a. We can also calculate the expected probability of packet loss over the entire period



(a) $\sigma = 10$ and v = 1 m/s



(b) $\sigma = 5$ and v = 1 m/s

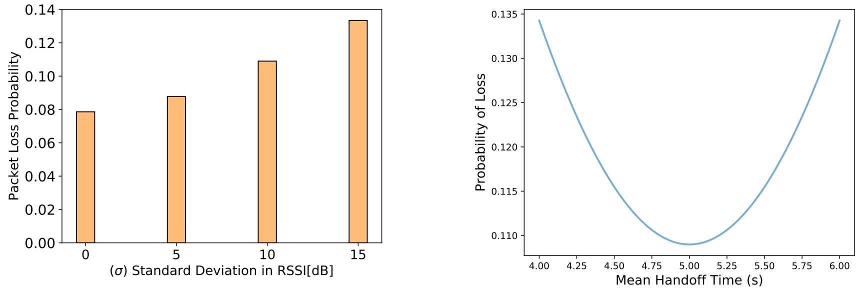


(c) $\sigma = 10$ and v = 2 m/s

Figure 9.6: Probability of packet loss, disconnections of individual links, and handoff in time. Comparing figures (**a**,**b**), we can observe higher packet loss probability stemming from fluctuations in the link quality. Comparing Figures (**a**,**c**), the higher packet loss probability is stemming from the increased velocity.

$(E(P_{Loss}))$ as depicted in Figure 9.7a to better see the increasing trend as a function of path loss variance.

On the other hand, a vital parameter in the model is the expected instant of time when the route-projection packet is received (T_{rx}). Here we assume that once the route-projection packet is received, the nodes omit the old routing state from its neighbor table and insert the new one with negligible delay. We assume that T_{rx} follows a normal distribution and its mean value is interpreted as Mean Handoff Time. P_{Loss} for increasing Mean Handoff Time is illustrated in Figure 9.7b. This figure shows that scheduling the handoff either too early or too late increases the probability of packet loss and the best time for receiving the route-injection packet is when the mobile node is placed at an equal distance from both SNs.



(a) Loss probability with respect to standard deviation in RSSI

(b) Loss probability with respect to expected T_{rx}

Figure 9.7: As illustrated in (**a**) with a higher variance of RSSI, it is expected to experience more packet loss in the system as the links are less stable. In (**b**), the packet loss probability with respect to the expected time for receiving the route-injection packet or T_{rx} (or Mean handoff time) is illustrated. The minimum packet loss happens when the packet is expected to be received in the middle of the trajectory (t = 5 for the scenario with 10 m distance).

9.4.3 Relation between the Analytical Evaluations and Simulations

To verify the mathematical abstractions that were presented in this section, it is both useful and common to validate the analytical results with simulations. Here we give a brief explanation of the simulations but the detailed explanation of the simulation environment can be found in the next section. The experiment includes 25 randomly placed anchors and 1 MN moving in a random trajectory.

The simulations showed the packet loss increases by path loss variance as illustrated in Figure 9.8a. The simulation results are aligned with the probability of packet loss in the analytical evaluation in Figure 9.7a.

As depicted in Figure 9.8b, increasing the velocity of the MN also impacts the reception ratio of the data packets in the same way as the analytical evaluations suggested (in Figure 9.6c).

9.5 Performance Evaluation

This section focuses on evaluating the performance of the SDMob. We have conducted a set of simulations based on Contiki's native emulator, Cooja [47]. The SDN controller has been implemented using a Linux machine with a Python-based filter, which connects to a C-based Contiki border router. For the IoT RPL/6LoWPAN network, we rely on the Contiki-NG/COOJA simu-

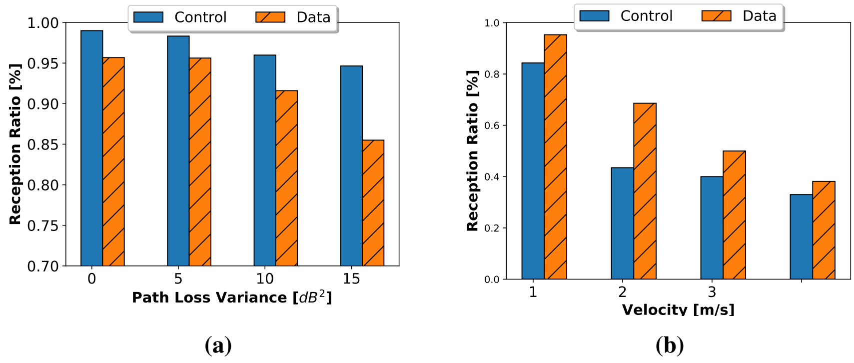


Figure 9.8: Simulation results for increasing (**a**) path loss variance and (**b**) velocity.

lation environment [48]. Contiki-NG is an open-source embedded operating system, which is easily portable to commodity hardware. We have chosen the well-known Sky platform for the nodes as it is very constrained (in terms of memory), however, the results have also been verified using a more recent Zolertia platform. Sky motes are using an MSP430 F1611 micro-controller featuring 10 kB of RAM and 48 kB of flash memory. Zolertia motes run on an MSP430 F2617 MCU with 8 KB of RAM and 92 KB of flash memory. Both motes communicate using Chipcon's IEEE 802.15.4 compliant CC2420 Radio working in the 2.4 GHz ISM band. The radio model incorporated in Cooja is the Distance Loss mode of the Unit Disk Graph Medium (UDGM). This model considers two circles around each mote that define the distance in which the packets can be received or can interfere with other transmissions.

Table 9.3 highlights the selected configuration for the simulations. The following metrics have been extracted for SDMob.

- *Position estimation accuracy:* is the difference between the estimated position and the ground truth and is expressed by Root Mean Squared Error (RMSE) which is a measure that represents the quadratic mean of the error and penalizes the larger errors.

- *Handoff delay:* Measuring handoff delay for SDMob is non-trivial as the start time of handoff is not defined and SDMob may perform a handoff in a proactive way when the previous link is still active. To define an upper bound for handoff delay, we assume the time that a link's quality drops below a lower threshold to be the start time of handoff.

- *End-to-end delay:* The time for data packets to reach the destination (sink node) from MN.

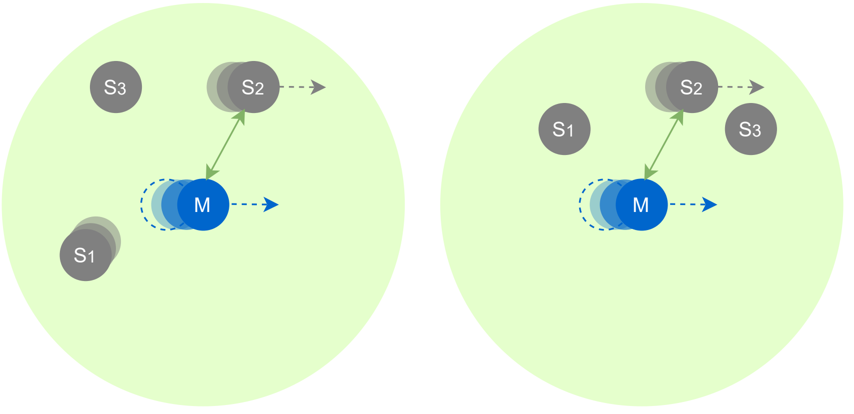
- *Packet Delivery Ratio:* The ratio of received packets (in the sink) to the transmitted packets.
- *Communication Overhead:* The accumulated number of bytes for control packets.

Table 9.3: Simulation parameters

Parameter	**Value**
Network Simulator	Cooja under Contiki-ng
Radio model	UDGM
Number of anchor nodes	10, 20, 30, 40, 50
Simulation Area	20×20m
Transmission range	5m
Simulation time	300 s
Initialization time	300 s
Beacon Interval	1 s
I_{min}	2^{12}
I_{max}	2^{20}
Buffer Timer	500 ms
Congestion Timer	0–50 ms (uniform distribution)

9.5.1 An Overview of the ARMOR

ARMOR [26] is one of the most recent mobility solutions integrated within RPL, where it shows better performance compared with MA-RPL [23]. Thus, we picked ARMOR as the benchmark to compare with SDMob. Authors of ARMOR argue that the mobility solutions that are based on the RSSI routing metric fail to select the most stable routes. Hence the key idea behind ARMOR is in introducing a new routing metric, called Time-to-Reside (TTR) that estimates for how long each candidate parent is available. Each node embeds its GPS position and velocity in a DIO packet and broadcasts it. The neighbors exploit this information to calculate the TTR. Using this metric, ARMOR also proposes a parent selection mechanism. ARMOR modifies the Trickle algorithm to have short and constant intervals between DIO packets. ARMOR also allows mobile nodes to advertise the DODAG and act as parents, which is a reasonable design choice for a network with many mobile nodes. The fact that mobile nodes can be set as parents is inevitable in a topology with many mobile nodes and it may require more frequent DIO intervals for all the nodes. In



(a) (b)

Figure 9.9: Two examples of ARMOR algorithm, (**a**) when mobile node *M* wisely selects *S2* as its parent since it provides the most long-lasting connection. RPL would have chosen *S1* by only considering the short-term link quality, and (**b**) depicts a scenario, where ARMOR is less effective as *M* ignores the stable (fixed) parents (*S1, S3*) and sticks to *S2*.

the same way, ARMOR allows MNs to serve as parents with the cost of transmitting frequent update packets. The interval for sending the control beacons in SDMob is tunable according to the physical speed of the mobile node.

ARMOR, like some of its counterparts, assumes that nodes are equipped with GPS modules, by which the location of the nodes is fed to the routing protocol. First, GPS systems are not practical in many applications because they perform poorly in an indoor environment. It is also possible to integrate other localization methods such as fingerprinting or triangulation, but the impact of its overhead and accuracy on the routing protocol needs to be investigated.

Figure 9.9a highlights a motivational scenario for ARMOR where it can do better than baseline RPL. Using the TTR metric, node *M* selects node *S2* as its parent since it provides a long-lasting connection. On the other hand, in Figure 9.9b, ARMOR neglects the more stable links provided by *S1* and *S3* and sticks to the *S2* with long-lasting but lower quality link.

9.5.2 Scaling to Multiple Mobile Nodes

SDMob lies in a spectrum in which one extreme is RPL with its scalability to thousands of nodes and the other extreme are the protocols specifically designed for mobile ad-hoc networks such as AODV. SDMob is designed for IoT applications that take advantage of the static multi-hop infrastructure and hence it can support a limited number of MNs.

Figure 9.10 shows the comparison of the evaluation metrics for SDMob and RPL in a scenario with 25 randomly placed anchor nodes and a number of MNs moving with Truncated Levy Walk mobility pattern (for detail on mobility patterns see Section 9.5.4). From the results, we observe that although RPL shows an increasing pattern in packet delivery ratio (PDR), the proposed SDMob remarkably outperforms RPL. RPL performs better in terms of PDR for an increasing number of MNs since the Trickle algorithm resets its interval, which leads to more frequent DIO packet transmissions that congest the network. This also justifies the rise in average E2E delay for RPL. The congestion caused by the increasing MNs affects the accuracy of localization that happens due to the loss of a significant number of RSSI reports. However, the particle filter showed a good ability to converge again after not receiving RSSI reports, and the impact on localization accuracy was tolerable. The difference in the upper bound of handoff delay was negligible (100 milliseconds). Figure 9.11 shows SDMob's communication overhead in comparison with standard RPL for increasing number of mobile nodes.

9.5.3 Scaling to Networks with High Density or Hop-Distance

The localization algorithm works based on the RSSI measurements, thereby being spotted in an area lacking enough anchors to retrieve this information is fatal for the filter. On the other hand, there may be scenarios with too many anchors or the anchors are so far that the RSSI reports do not reach the controller in time.

SDMob's anchor nodes inherit RPL's design choices when it comes to handling high data rate traffic. Once an MN enters an area with a large number of anchors, it will congest the network and hence the local links will measure a higher ETX. Most standard-compliant implementations of RPL reset the DIO interval when there is a significant change in the rank value assuming that it has been caused by a lossy link. This leads to more frequent transmissions of DIO packets and further congesting the network. To avoid the aforementioned scenario, we have employed the congestion timer. The simulated set of scenarios consists of one MN moving in a linear trajectory and the anchors are manually positioned to keep the number of neighbors constant over time as depicted in Figure 9.12. Figure 9.13 shows the performance of SDMob in networks with an increasing number of neighbors. For up to 5 neighbors, we can see an increasing trend in the PDR but for 6 neighbors the network gets so congested that a sharp drop in the PDR is noticed both for control and data traffic. Localization and E2E delay also follow the same pattern.

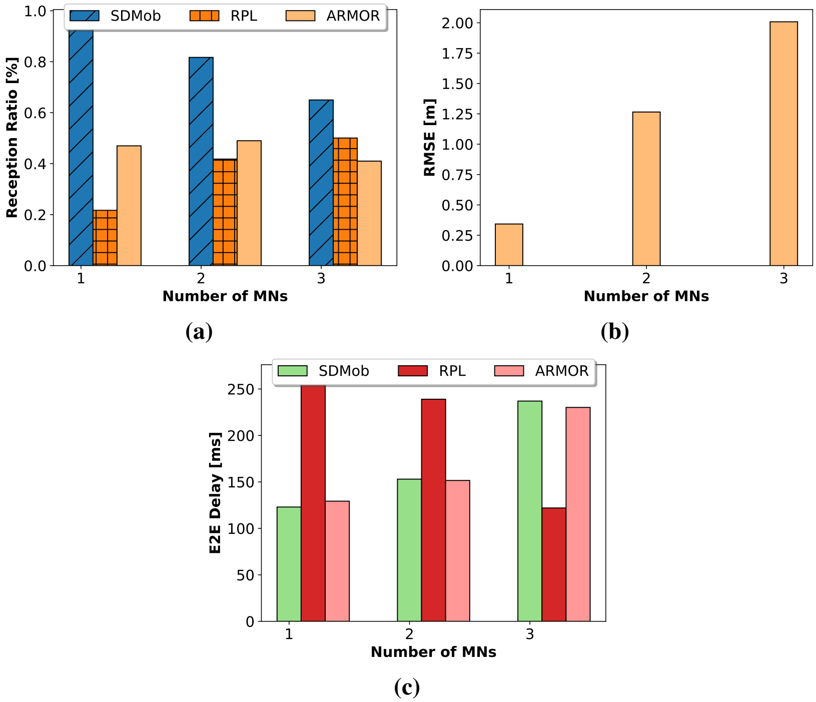


Figure 9.10: Simulation results for increasing number of MNs with different solutions—(**a**) PDR; (**b**) localization error; and (**c**) end-to-end delay.

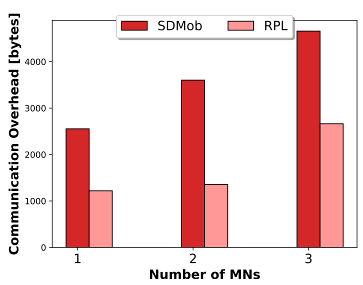


Figure 9.11: Communication overhead (in bytes) for increasing number of MNs with different solutions.

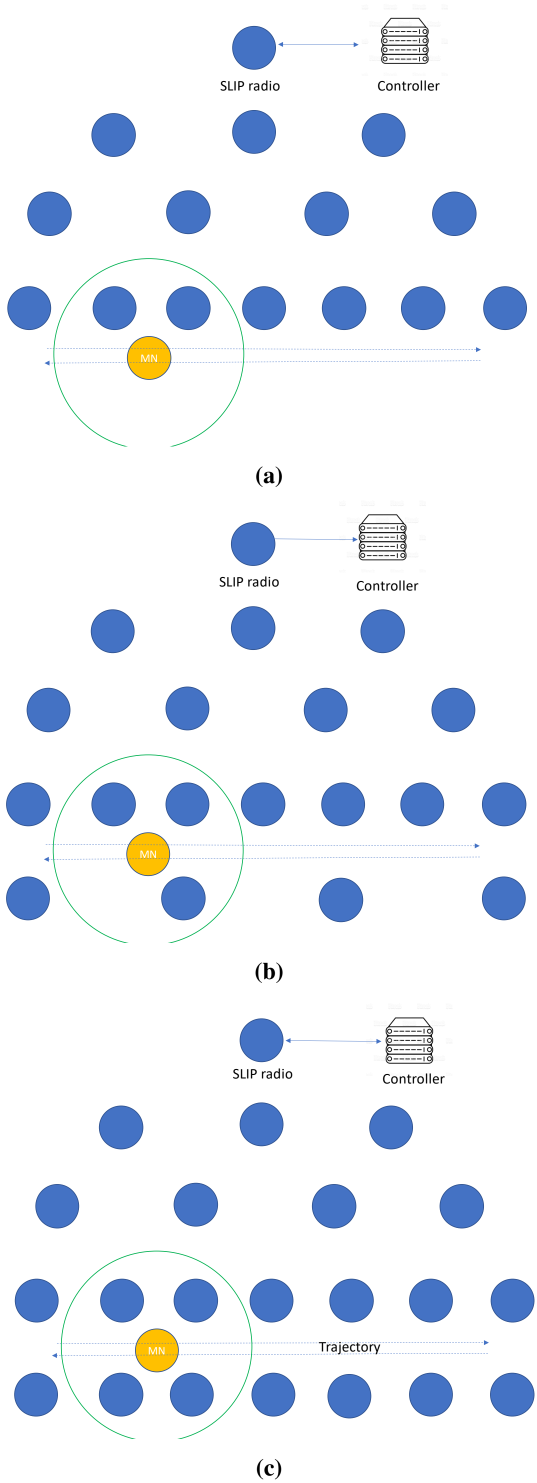


Figure 9.12: Simulation scenarios when increasing number of neighbors of a MN: (**a**) with 2 neighbors, (**b**) with 3 neighbors, and (**c**) with 4 neighbors.

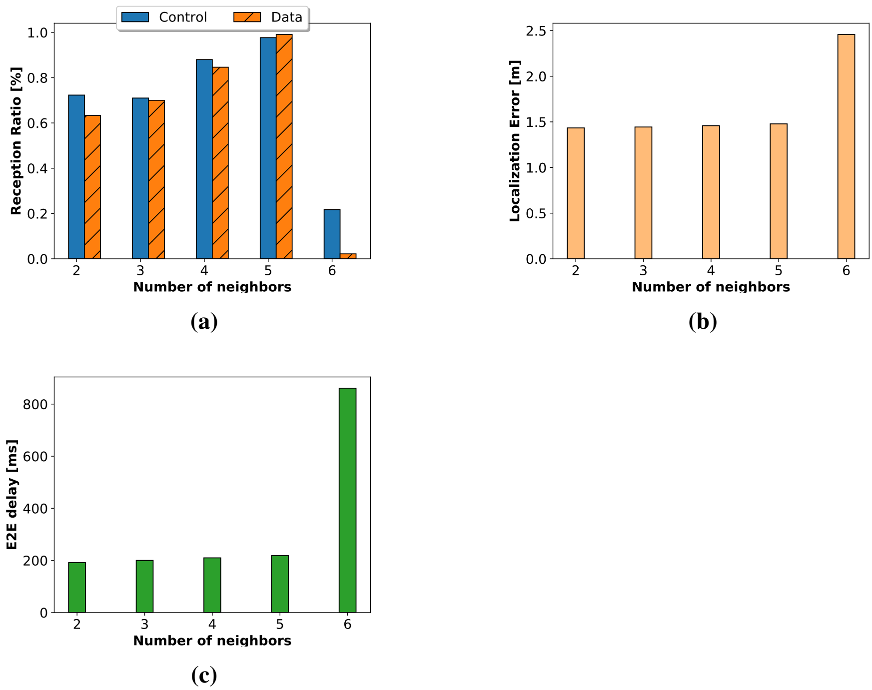
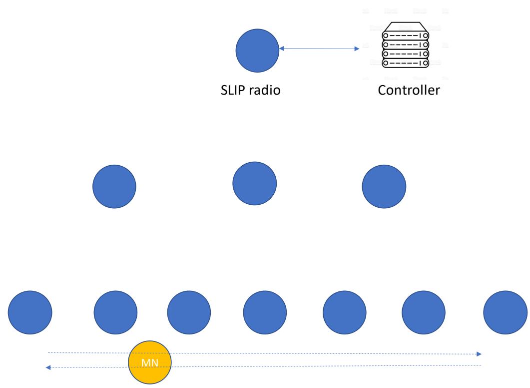


Figure 9.13: Simulation results for increasing number of neighbors with SDMob—(a) PDR; (b) localization error; and (c) end-to-end delay.

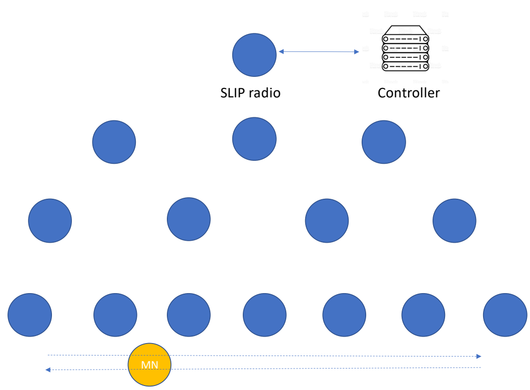
Another parameter that changes in scale is the hop distance from the MN to the controller. To examine the performance of SDMob in this regard, we manually placed the anchors to keep the hop distance of MN constant over time as illustrated in Figure 9.14 and experimented with different distances. As illustrated in Figure 9.15, the PDR and localization are more stable with increasing hop distance. However, there is a continual growth in E2E delay and handoff delay.

9.5.4 Mobility Patterns

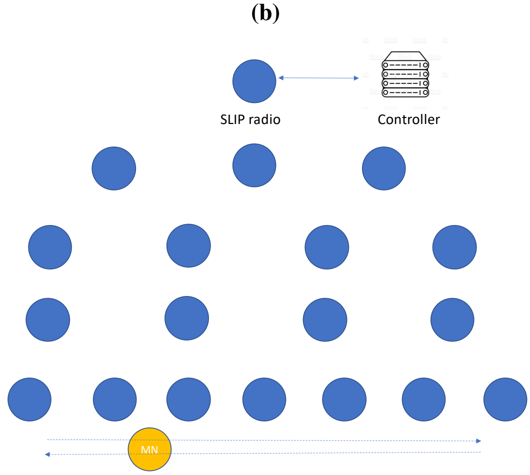
The previous tests required a deterministic configuration that could only be performed using a linear moving trajectory and a chosen topology. Now we shift the focus on a random placement of nodes and trajectory. We utilized an open-source library called pymobility [49] to create some of the most well-known mobility traces in the field. The number of neighbors and hop-distance depends on the mobility pattern and changes in time. There are 25 anchors and one MN in each of the scenarios below.



(a)



(b)



(c)

Figure 9.14: Simulation scenarios when increasing distance between MN and controller, (**a**) with 3-hop distance, (**b**) with 4-hop distance, and (**c**) with 5-hop distance.

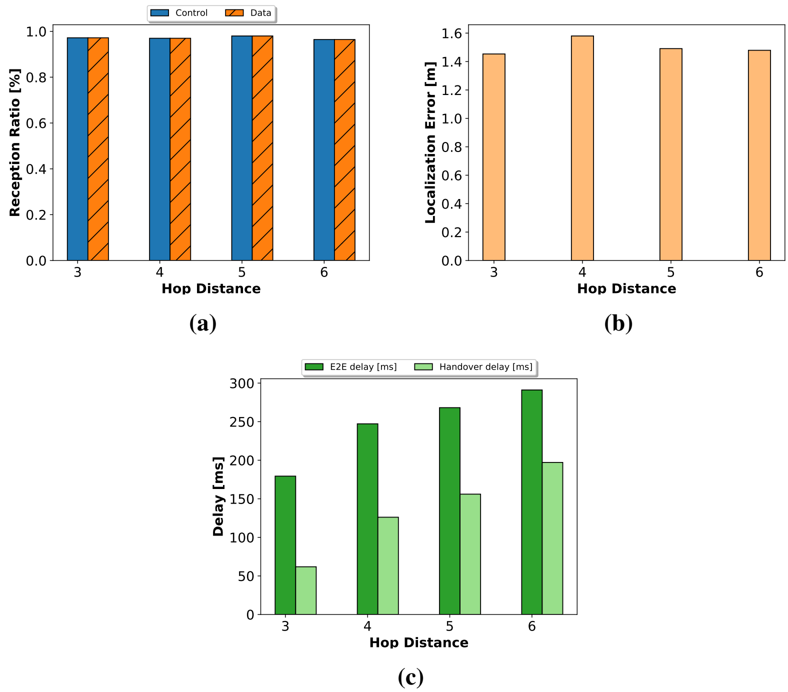


Figure 9.15: Simulation results for increasing distance between MN and controller with SDMob—(**a**) PDR; (**b**) localization error; and (**c**) end-to-end delay.

Random Way Point (RWP).

RWP is a synthetic mobility model in which MNs choose a random point in the simulation area and start moving toward it with a randomly chosen velocity. In random waypoint, nodes tend to be spotted in the middle of the simulation area. Since the newly selected point is random, sharp turns may happen that are not realistic. Another phenomenon called density wave [50] is that the number of neighbors for each MN fluctuates considerably. As the MN passes through the center of the simulation area, where it is usually more crowded (by both static and MNs), it may further converge the network. On the other hand, when the MN is spotted closer to the edges, it may be subject to packet losses in blind spots.

Random Direction Model (RDM).

In RDM, MNs randomly choose a direction until it reaches the boundary of the area, and after a pause, chooses a new direction as depicted in Figure 9.16a.

If the number of MNs is high, this will decrease the probability of congestion in the center as the MNs tend to be spotted at the edges of the area, and thus it is claimed that RDM is known to be unaffected density wave problem. This paper aims at extending RPL to support a limited number of MNs. As a result, the fact that nodes are more often spotted at the edges does not considerably decrease the congestion, and RDM is even expected to reduce the number of RSSI measurements that arrive at the controller and are fed to the filter. The filter works reliably when it receives at least 3 measurements. The effect of the number of neighbors has been studied in the previous subsection.

Gauss Markov Model (GMM).

GMM is a memory-based mobility pattern in which the temporal dependency of the nodes can be defined using a parameter α (between 0 and 1). Higher values of α imply higher dependency and less harsh turns and speed changes. This leads to more realistic mobility patterns and configurable randomness.

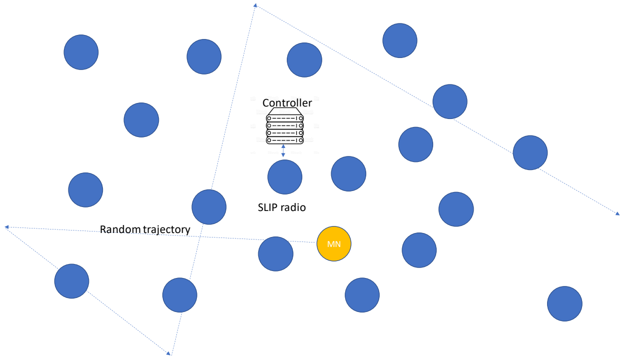
Truncated Levy Walk (TLW).

It has been claimed that human mobility has the same characteristics as Levy Walks that follows the heavy-tailed Levy distribution [51]. This leads to a number of short flights followed by a long flight and fewer harsh turns as depicted in Figure 9.16b. Therefore it is expected for the filter to perform better under this mobility pattern.

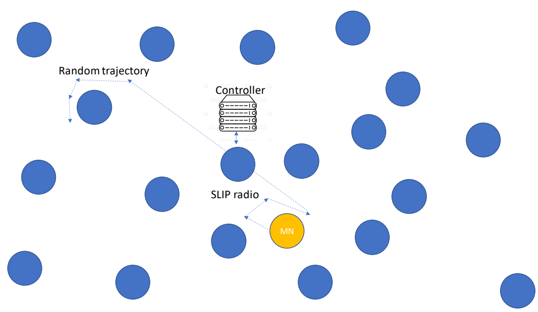
Figure 9.17 compares the evaluation metrics for different mobility patterns. The distribution of velocity is different but we kept the same mean velocity for fairness. The PDR reached a peak during the simulation with TLW since the harsh turns are minimized in this mobility pattern and there are a lot of short flights that reduce the number of required handoffs. In RDM, the data traffic is not as well-received as the control traffic. This is caused by the fact that RDM challenges the localization algorithm with unpredictable churns. For RWP and RDM, the localization error is higher than TLW and GMM. This is due to the higher randomness in direction and velocity in these patterns. E2E delay for RDM is shown to be higher because of the MN residing in the edges of the simulation area.

9.5.5 Velocity

Another decisive parameter is the velocity of the MN. Target applications mandate supporting the physical speed of humans (for healthcare applications) averaging about 5 km/h or 1.34 m/s. For industrial applications, many machines



(a)



(b)

Figure 9.16: Simulation scenarios with different mobility patterns: (**a**) with RDM mobility pattern and (**b**) with TLW mobility pattern.

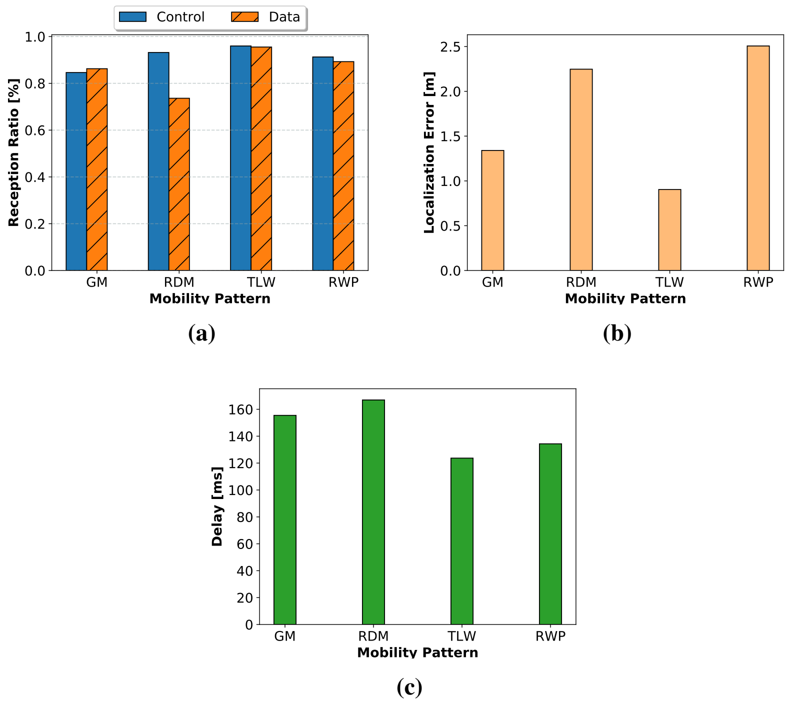
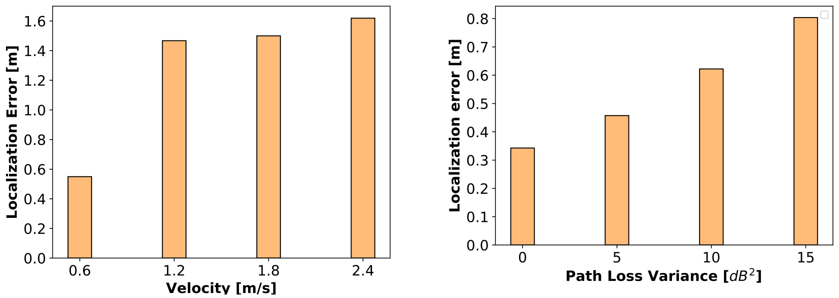


Figure 9.17: Simulation results for different mobility patterns with SDMob—(**a**) PDR; (**b**) localization error; and (**c**) end-to-end delay.

such as forklifts are capable of speeds over 22 km/h but regulatory agencies such as Occupational Safety and Health Administration recommend a speed of 5 km/h for typical indoor environments [52]. We have evaluated the performance of the system with speeds up to 2.4 m/s.

We have tested a group of scenarios in which a single MN roams around according to the TLW model with its speed averaging to the values between 0.6 m/s to 2.4 m/s. What stands out in Figure 9.18a is the steady rise in localization error with increasing velocity. In the previous section in Figure 9.8b, we observed a general decreasing pattern of PDR with increasing velocity of the MN.

The loss in packet delivery ratio with an increased physical speed of MN stems from both (i) lower accuracy of the filter and (ii) higher probability for delayed reception of the SET packet. It is possible to increase the supported maximum speed of the MN. At a higher velocity, it is more challenging to keep the accuracy of the filter unless by transmitting more frequent control beacons. A more frequent control beacon is not always desirable since it im-



(a) **(b)**

Figure 9.18: Average localization error as a function of (**a**) MN's velocity, and (**b**) path loss variance.

poses a higher communication overhead. This higher overhead may avoid the boosted localization accuracy from improving the PDR. We consider dynamically changing this beaconing interval for future work.

9.5.6 Path Loss Variance

For changing this parameter, we had to enhance the radio model in Cooja to support a zero-mean normal distribution. Figure 9.18b reveals that if the link qualities are fluctuating more (due to the environment characteristics) it gets more difficult to track the MN so SDMob shows a slight escalation in localization error.

In Section 9.4, the degrading impact of path loss variance on reception ratio in the simulations was compared with the analytical evaluations in Figure 9.8a. It is worth mentioning that although the reception ratio for control traffic is not affected much, the data traffic's PDR is about 10 percent less than control traffic. This can be explained by the fact that the RSSI fluctuations contribute to packet loss.

9.6 Conclusions

To address the low reliability of the RPL protocol in mobile IoT applications, we have proposed SDMob. In this proposed architecture, an edge device collaborates with the distributed nodes to provide seamless handoff for the mobile nodes. However, new challenges arise, such as delivering real-time link quality reports to the controller as well as the downward packets to install root-initiated routes. SDMob addresses these challenges by employing a lightweight controller that tunes its operation for a constrained environment. Having the controller deployed, the computation-intensive tasks can be offloaded to the controller to benefit from the global view and resources available at the centralized controller.

The results show that SDMob significantly improves RPL and outperforms state-of-the-art ARMOR with close to 100 percent PDR, with reasonable and adjustable overhead in scenarios with a varying number of mobile nodes. Given the requirements of the applications, SDMob by design aims at reliably maintaining the connectivity of a limited number of mobile nodes since with increasing mobile nodes the control traffic increases as well. The solutions in the literature that aim at a higher number of mobile nodes usually make a compromise by less reliable communication for mobile nodes. Through analytical evaluations, we analyzed the behavior of the system when exposed to increasing path loss variance in the radio environment, and velocity of the mobile node which was also verified by simulations. We extended the simulation results to networks with varying densities of neighbors, the distance of the mobile node from the controller (in terms of the number of hops), and mobility patterns.

For future work, we consider employing redundant controllers to avoid a single point of failure in the control layer, although, in the event of a failure in the controller, RPL resumes its normal operation. Another interesting direction is applying machine learning methods to either localize the mobile node and predict its link quality in time, or automatically optimize different system parameters such as beacon interval and congestion and buffer timers.

Bibliography

[1] Bardia Safaei, Aliasghar Mohammadsalehi, Kimia Talaei Khoosani, Saba Zarbaf, Amir Mahdi Hosseini Monazzah, Farzad Samie, Lars Bauer, Jörg Henkel, and Alireza Ejlali. Impacts of mobility models on rpl-based mobile iot infrastructures: An evaluative comparison and survey. *IEEE Access*, 8:167779–167829, 2020.

[2] Philip Levis, Thomas Clausen, Jonathan Hui, Omprakash Gnawali, and JeongGil Ko. The trickle algorithm. *Internet Engineering Task Force, RFC6206*, 2011.

[3] Anup Kumar Paul and Takuro Sato. Localization in wireless sensor networks: A survey on algorithms, measurement techniques, applications and challenges. *Journal of sensor and actuator networks*, 6(4):24, 2017.

[4] Simo Särkkä. *Bayesian filtering and smoothing*, volume 3. Cambridge University Press, 2013.

[5] Li Li, Wang Yang, and Guojun Wang. Intelligent fusion of information derived from received signal strength and inertial measurements for indoor wireless localization. *AEU - International Journal of Electronics and Communications*, 70(9):1105–1113, 2015.

[6] Olivier Cappe, Simon J. Godsill, and Eric Moulines. An overview of existing methods and recent advances in sequential Monte Carlo. *Proceedings of the IEEE*, 95(5):899–924, 2007.

[7] Marc Barcelo, Alejandro Correa, Jose Lopez Vicario, Antoni Morell, and Xavier Vilajosana. Addressing Mobility in RPL with Position Assisted Metrics. *IEEE Sensors Journal*, 16(7):2151–2161, 2016.

[8] Maha Bouaziz, Abderrezak Rachedi, and Abdelfettah Belghith. EKF-MRPL: Advanced mobility support routing protocol for internet of mobile things: Movement prediction approach. *Future Generation Computer Systems*, 93:822–832, 2019.

[9] Allan J.R. Gonçalves, Ricardo A.L. Rabêlo, Joel J.P.C. Rodrigues, and Luís M.L. Oliveira. A mobility solution for low power and lossy networks using the LOADng protocol. *Transactions on Emerging Telecommunications Technologies*, 31(November 2019):1–24, 2020.

[10] Antonio Ramón Jiménez Ruiz and Fernando Seco Granja. Comparing ubisense, bespoon, and decawave uwb location systems: Indoor performance analysis. *IEEE Transactions on instrumentation and Measurement*, 66(8):2106–2117, 2017.

[11] Hlabishi I Kobo, Adnan M Abu-Mahfouz, and Gerhard P Hancke. A survey on software-defined wireless sensor networks: Challenges and design requirements. *IEEE access*, 5:1872–1899, 2017.

[12] Iliar Rabet, Shunmuga Priyan Selvaraju, Mohammad Hassan Adeli, Hossein Fotouhi, Ali Balador, Maryam Vahabi, Mário Alves, and Mats Björkman. Pushing iot mobility management to the edge: Granting rpl accurate localization and routing. In *2021 IEEE 7th World Forum on Internet of Things (WF-IoT)*, pages 338–343. IEEE, 2021.

[13] Hossein Fotouhi, Daniel Moreira, and Mário Alves. mRPL: Boosting mobility in the Internet of Things. *Ad Hoc Networks*, 26:17–35, 2015.

[14] Omprakash Gnawali and Philip Levis. The minimum rank with hysteresis objective function. *RFC 6719*, page 13, 2012.

[15] P Thubert. Objective function zero for the routing protocol for low-power and lossy networks (rpl [online] available: https://tools. ietf. org/html/rfc6552. *Accessed: April*, 2017.

[16] Afonso Oliveira and Teresa Vazao. Low-power and lossy networks under mobility: A survey. *Computer networks*, 107:339–352, 2016.

[17] Patrick Olivier Kamgueu, Emmanuel Nataf, Thomas Djotio Ndie, Patrick Olivier Kamgueu, Emmanuel Nataf, Thomas Djotio, and Ndie Survey. Survey on RPL enhancements : a focus on topology , security and mobility To cite this version : HAL Id : hal-01713247. *Computer Communications*, 2018.

[18] Yad Tahir, Shusen Yang, and Julie McCann. Brpl: Backpressure rpl for high-throughput and mobile iots. *IEEE Transactions on Mobile Computing*, 17(1):29–43, 2017.

[19] Michael J Neely. Stochastic network optimization with application to communication and queueing systems. *Synthesis Lectures on Communication Networks*, 3(1):1–211, 2010.

[20] Hossein Fotouhi, Daniel Moreira, Mário Alves, and Patrick Yomsi. mRPL+: A mobility management framework in RPL/6LoWPAN. *Computer Communications*, 104:34–54, 2017.

[21] Bin Tian, Kun Mean Hou, Hongling Shi, Xing Liu, Xunxing Diao, Jianjin Li, Yibo Chen, and Jean-Pierre Chanet. Application of modified RPL under VANET-WSN communication architecture. In *2013 international conference on computational and information sciences*, pages 1467–1470. IEEE, 2013.

[22] Jihong Park, Ki Hyung Kim, and Kangseok Kim. An algorithm for timely transmission of solicitation messages in RPL for energy-efficient node mobility. *Sensors (Switzerland)*, 17:1–21, 2017.

[23] Sarumathi Murali and Abbas Jamalipour. Mobility-aware energy-efficient parent selection algorithm for low power and lossy networks. *IEEE Internet of Things Journal*, 6(2):2593–2601, 2018.

[24] Emilio Ancillotti, Carlo Vallati, Raffaele Bruno, and Enzo Mingozzi. A reinforcement learning-based link quality estimation strategy for RPL and its impact on topology management. *Computer Communications*, 112:1–13, 2017.

[25] Harith Kharrufa, Hayder Al-Kashoash, and Andrew H Kemp. A game theoretic optimization of rpl for mobile internet of things applications. *IEEE Sensors Journal*, 18(6):2520–2530, 2018.

[26] Aliasghar Mohammadsalehi, Bardia Safaei, Amir Mahdi Hosseini Monazzah, Lars Bauer, Jorg Henkel, and Alireza Ejlali. ARMOR: A Reliable and Mobility-aware RPL for Mobile Internet of Things Infrastructures. *IEEE Internet of Things Journal*, pages 1–1, jun 2021.

[27] H Farag, P Österberg, and M Gidlund .... RMA-RP: A Reliable Mobility-Aware Routing Protocol for Industrial IoT Networks. In *IEEE Global Conference on Internet of Things (GCIoT)*, 2019.

[28] Ramviyas Parasuraman, Petter Oegren, and Byung Cheol Min. Kalman Filter Based Spatial Prediction of Wireless Connectivity for Autonomous Robots and Connected Vehicles. *IEEE Vehicular Technology Conference*, 2018-Augus, 2018.

[29] Thomas Clausen, Jiazi Yi, and Axel Colin De Verdiere. LOADng: Towards aodv version 2. In *2012 IEEE Vehicular Technology Conference (VTC Fall)*, pages 1–5. IEEE, 2012.

[30] L. Mihaylova, D. Angelova, S. Honary, D. R. Bull, C. N. Canagarajah, and B. Ristic. Mobility tracking in cellular networks using particle filtering. *IEEE Transactions on Wireless Communications*, 6(10):3589–3599, 2007.

[31] Fabian Fernando Jurado Lasso, Letizia Marchegiani, Jesus Fabian Jurado, Adnan Abu Mahfouz, and Xenofon Fafoutis. A survey on software-defined wireless sensor networks: Current status, machine learning approaches and major challenges. *PrePrint*, 2021.

[32] Zohaib Latif, Kashif Sharif, Fan Li, Md Monjurul Karim, Sujit Biswas, and Yu Wang. A comprehensive survey of interface protocols for software defined networks. *Journal of Network and Computer Applications*, 156:102563, 2020.

[33] Pascal Thubert, R.A. Jadhav, and Gillmorem M. Root initiated routing state in RPL draft-ietf-roll-dao-projection-09, 2019.

[34] Iliar Rabet, Hossein Fotouhi, Maryam Vahabi, Mats Björkman, and Mario Alves. Rpl-rp: Rpl with route projection for transversal routing. In *IEEE World Forum IoT 2021*, 2021.

[35] George Violettas, Sophia Petridou, and Lefteris Mamatas. Routing under Heterogeneity and Mobility for the Internet of Things: A Centralized Control Approach. In *2018 IEEE Global Communications Conference, GLOBECOM 2018 - Proceedings*, 2018.

[36] Tryfon Theodorou and Lefteris Mamatas. Sd-miot: A software-defined networking solution for mobile internet of things. *IEEE Internet of Things Journal*, 8(6):4604–4617, 2020.

[37] Michael Baddeley, Reza Nejabati, George Oikonomou, Mahesh Sooriyabandara, and Dimitra Simeonidou. Evolving SDN for low-power iot networks. In *2018 4th IEEE Conference on Network Softwarization and Workshops (NetSoft)*, pages 71–79. IEEE, 2018.

[38] Laura Galluccio, Sebastiano Milardo, Giacomo Morabito, and Sergio Palazzo. SDN-WISE: Design, prototyping and experimentation of a stateful sdn solution for WIreless SEnsor networks. In *2015 IEEE Conference on Computer Communications (INFOCOM)*, pages 513–521. IEEE, 2015.

[39] Lucia Lo Bello, Alfio Lombardo, Sebastiano Milardo, Gaetano Patti, and Marco Reno. Experimental assessments and analysis of an sdn framework to integrate mobility management in industrial wireless sensor networks. *IEEE Transactions on Industrial Informatics*, 16(8):5586–5595, 2020.

[40] Federico Orozco-Santos, Víctor Sempere-Payá, Javier Silvestre-Blanes, and Teresa Albero-Albero. Multicast scheduling in sdn wise to support mobile nodes in industrial wireless sensor networks. *IEEE Access*, 9:141651–141666, 2021.

[41] JS Mertens, GM Milotta, P Nagaradjane, and Giacomo Morabito. Sdn-(uav) ise: Applying software defined networking to wireless sensor networks with data mules. In *2020 IEEE 21st International Symposium on" A World of Wireless, Mobile and Multimedia Networks"(WoWMoM)*, pages 323–328. IEEE, 2020.

[42] Hossein Fotouhi, Maryam Vahabi, Iliar Rabet, Mats Björkman, and Mário Alves. MobiFog: Mobility Management Framework for Fog-

assisted IoT Networks. In *IEEE Global Conference on Internet of Things GCIoT'19, 04 Dec 2019, Dubai, United Arab Emirates*, 2019.

[43] Pascal Thubert and M Richardson. Routing for RPL leaves. *Work in Progress, draft-thubert-roll-unaware-leaves-05*, 2018.

[44] Marco Zuniga and Bhaskar Krishnamachari. Analyzing the transitional region in low power wireless links. In *2004 First Annual IEEE Communications Society Conference on Sensor and Ad Hoc Communications and Networks, 2004. IEEE SECON 2004.*, pages 517–526. IEEE, 2004.

[45] Jeroen D Hol, Thomas B Schon, and Fredrik Gustafsson. On resampling algorithms for particle filters. In *2006 IEEE nonlinear statistical signal processing workshop*, pages 79–82. IEEE, 2006.

[46] Iliar Rabet, Shunmuga Priyan Selvaraju, Hossein Fotouhi, Maryam Vahabi, and Mats Bjorkman. Poster: Particle Filter for Handoff Prediction in SDN-based IoT Networks. In *EWSN 2020*, 2019.

[47] Fredrik Osterlind, Adam Dunkels, Joakim Eriksson, Niclas Finne, and Thiemo Voigt. Cross-level sensor network simulation with cooja. In *Proceedings. 2006 31st IEEE conference on local computer networks*, pages 641–648. IEEE, 2006.

[48] Fredrik Österlind. A sensor network simulator for the contiki os. *SICS Research Report*, 2006.

[49] André Panisson. pymobility v0.1 - python implementation of mobility models, 2014.

[50] Elizabeth M Royer, P Michael Melliar-Smith, and Louise E Moser. An analysis of the optimum node density for ad hoc mobile networks. In *ICC 2001. IEEE International Conference on Communications. Conference Record (Cat. No. 01CH37240)*, volume 3, pages 857–861. IEEE, 2001.

[51] Injong Rhee, Minsu Shin, Seongik Hong, Kyunghan Lee, Seong Joon Kim, and Song Chong. On the levy-walk nature of human mobility. *IEEE/ACM transactions on networking*, 19(3):630–643, 2011.

[52] Occupational Safety, DC. Health Administration, Washington, United States. Occupational Safety, Health Administration, Labor Dept.(US) Occupational Safety, Health Administration, et al. *Training requirements in OSHA standards and training guidelines*. Number 2254. Occupational Safety & Health Administration, 1998.