MobiFog: Mobility Management Framework for Fog-assisted IoT Networks

Hossein Fotouhi*, Maryam Vahabi*, Iliar Rabet*, Mats Björkman*, and Mário Alves†
*School of Innovation, Design and Engineering, Mälardalen University, Västerås, Sweden
†Politécnico do Porto, ISEP/IPP, Porto, Portugal
Email: *{hossein.fotouhi, maryam.vahabi, iliar.rabet, mats.bjorkman}@mdh.se, †mjf@isep.ipp.pt

***Abstract*—Mobility is becoming a challenging issue in upcoming IoT applications, where it is crucial to employ mobile entities. Patients with sensors attached to their body in health monitoring application, AGVs in industrial monitoring and factory automation applications, cars with several sensing devices in vehicular applications are a few examples of use cases with the need for mobile nodes. 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.**

***Index Terms*—Mobility Management, IoT, seamless handoff, Fog computing. IP protocol, 6loWPAN, RPL, TSCH, Contiki, Simulation, Network Performance, Probabilistic model,...**

I. Introduction

The current trends in the Internet of Things (IoT) show a growing need for flexibility. Most IoT applications have been building on homogeneous and static network infrastructures, while 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 topological changes, deriving from variations in the surrounding environment, such as electromagnetic noise, physical conditions (temperature, humidity, pressure), and moving entities (people, vehicles, animals, furniture, objects) [1].

Under such unreliable constrained wireless networks, messages from IoT devices may frequently be lost or delayed. Consequently, as constrained devices move around different wireless networks, its Internet Protocol (IP) connectivity may be frequently disrupted and power can be drained rapidly due to the massive amount of signaling messages for network discovery [2]. This can in turn result in the loss of important sensing data or imposing large delays for time-sensitive applications such as healthcare monitoring and factory automation. Moreover, existence of a heterogeneous network will increase probability of interference for devices operating in same frequency band. Beside physical mobility, interference will also impose topological changes (known as logical mobility) in such a heterogeneous IoT network, which in turn will degrade network performance.

Dealing with the dynamics of these heterogeneous IoT networks, involving mobile entities with resource constraints is very challenging. The Fog Computing paradigm is an architectural enabler for designing more flexible networks. Fog computing extends the Cloud-like services to the network edge to deal with IoT devices locally to carry out most of the storage, communication, control, configuration and management [3]. Thus, exploiting Fog Computing can in large extent improve the Quality-of-Service (QoS) in IoT networks.

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. It is common to keep the same rules that are set during the network design (pre-run-time) even upon high variations in network conditions at run-time. We believe that there is a strong need for devising a centralized network management scheme for IoT networks to optimize bandwidth, message scheduling and mobility management. Software-Defined Networking (SDN) may be a major enabler of Fog Computing, leveraging an efficient management of heterogeneous and mobile networks [4]. 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. In fact, SDN revolutionizes networking protocols and algorithms by providing higher level management in the system, while enhancing the quality of the service.

A typical mobility solution 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), which are basically more powerful nodes in terms of radio coverage, processing power and energy sustainability. For simplicity, 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 [5]–[7]. Using an SDN controller for mobility management will require the redesign of the handoff mechanism.

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 II presents the related works

on mobility management, Fog computing, SDN and network scheduling. Section III describes some of the handoff techniques that are employed in conventional IoT architectures. Section IV explains the problem with mobility solutions in conventional IoT architectures. Section V gives a general model of the proposed approach (MobiFog) to support seamless handoff mechanism.

II. Literature review

In this section, we overview state-of-the-art in the topics related to this paper - mobility management, Fog computing, software-defined networking and network scheduling. We identify existing gaps and key research questions to be answered when designing a Fog-based mobility mechanism.

Mobility management. A major challenge in the IoT applications is mobility support, which enables seamless and continuous connectivity of mobile nodes without loss of QoS. For instance, mobility management enables wireless physiological devices in health monitoring applications to remain connected to the Internet through low-power wireless technologies. This feature enables mobile nodes to dynamically connect to different APs in order to extend their coverage and provide acceptable QoS when moving. Furthermore, minimum network disconnections are a prerequisite for time sensitive applications which implies that for example the medical staff can receive reliable and timely information about patients under monitoring regardless of their physical activity status.

There have been many efforts on devising mobility management solutions for IoT and wireless sensor networks. We provided a generic handoff model [5], [8], followed by the integration of smart-HOP within the RPL routing protocol (mRPL [6] and mRPL+ [7]). Several other works focused on mobility support in commodity IoT protocols by enhancing 6loWPAN, RPL, and CoAP [9]–[12]. A recent survey paper collects some of the related works in this area [13]. 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.

Fog computing. To support the computational demand of real-time and delay-sensitive applications in large-scale IoT networks, a new computing paradigm named Fog computing has been introduced. In fact, Fog computing acts as an intermediate layer between the IoT devices and the Cloud data centers [14]. Several Fog Computing architectures have been proposed in the literature, which differ in terms of number of intermediate layers and devices. For instance, [15] considers a 4-layered architecture from the sensing to the Cloud data centers. The latency varies from milliseconds at the sensing level, reaching to seconds at the edge, and minutes at the intermediate computing nodes, and hours at the data centers. We believe that the Fog computing architecture is highly dependent on the application domain in terms of reliability and timeliness. Although Fog computing is usually known as just a new computing paradigm, it also represents new paradigms in communication, storage, and control architectures [16]. Fog computing networking explores how devices may talk to each other, despite intermittent global connectivity. It is of paramount importance to devise a Fog computing architecture for IoT applications considering their resource limitations.

Software Defined Networking. SDN [17], [18] is a new paradigm for communication networks that focuses on separating of the control plane from the data plane, and thus creating a flexible architecture that allows quick and easy configuration of the network. This ability is very useful in networks that should adapt to sudden changes, such as changes in network traffic or physical movement of nodes. OpenFlow [19] is the most prominent approach that implements SDN concept and offers a high flexibility in the routing protocol. There have been many efforts to design SDN controllers for IoT networks, specifically for sensor networks [20]–[22]. 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 [23]. 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 [24]. It is required to have a connection from the OpenFlow SDN controller to an existing IoT OS (e.g. Contiki OS). This enables employing the current IoT technologies in the Contiki OS while supporting the SDN controller.

Network scheduling. It is of paramount importance to investigate and employ a proper network scheduling paradigm for IoT networks. IoT applications are supposed to conduct various sensing and measurement activities, while some of them with real-time requirements. Network scheduling techniques provide solutions to schedule control and data packets over slots in a way to avoid collision in the network, while ensuring QoS. Mobility management requires extra signaling, which may occur at different levels, between APs and MN, between APs and the SDN controller, or between MN and the SDN controller. Accommodating all the required signaling messages within the network for supporting the mobility management is a challenging issue.

Different network scheduling schemes have been proposed for IoT applications in order to accommodate the traffic. However, these scheduling schemes lack the capability to meet the dynamics of the network imposed by either various traffic patterns or node mobility [25]. The Time-Slotted Channel Hoppint (TSCH) protocol combines three mechanisms: (i) time slotted access, (ii) multi-channel communication, and (iii) channel hopping [26]. The combination of all these features makes TSCH a suitable scheme for IoT applications with highly unreliable links. The slot frame indicates whether a node should transmit, receive or sleep. For each slot, the schedule dictates the node which neighbor to communicate with and on which channel offset. Using multi-channel communication, several nodes can communicate at the same time, which increases network capacity. The channel hopping mechanism allows nodes to change the communication channel between a predefined sequence of channels.

The IETF 6TiSCH Working Group is currently standardizing the mechanisms to run IPv6 on top of TSCH [27].

It also defines 6top as a sub-layer that enables neighbor-to-neighbor slot installation/removal. 6top can run one or several scheduling functions, which defines rules about when and how to add and remove slots at each node. TSCH orchestrates the medium access according to a communication schedule that indicates what should each node do in each slot and frequency channel. The IEEE 802.15.4 standard [28] only specifies how the MAC layer executes the schedule. However, it does not define how to establish it. Therefore, several scheduling algorithms have been proposed for TSCH networks. They can be broadlyN classified as either centralized [29]–[31] or distributed [32], [33] algorithms.

We believe that IoT networks based on a Fog Computing architecture may require a hybrid algorithm, where decisions, such as handoff decision (choosing best parent with proper time and frequency) are conducted at different levels (IoT, Fog and Cloud). An efficient scheduling scheme should avoid significant signaling overhead. Moreover, the standard protocol and current state-of-the-art focuses on fixed schedules, while we are envisioning an on-the-fly scheduling approach.

III. Background on handoff mechanisms

Previously, we have proposed hard and soft handoff approaches for conventional IoT networks, where these ideas are partially used in the MobiFog model. This section provides a summary on smart-HOP, mRPL and mPL+.

Smart-HOP. This algorithm has two main phases of Data Transmission and Discovery. Assume that a MN is in the range of three APs (AP_1, AP_2 and AP_3). We assume that the MN is attached to AP_1 and communicates with this AP with a reliable link in the Data Transmission Phase. The 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). These link quality metrics are the parameters that the radio measures upon signal reception.

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 reply beacons from the serving and neighboring APs in both Data Transmission and Discovery phases are unicast DIO packets. The bursts of beacons in the Discovery Phase are multicast DIS messages that request for getting a reply (DIO) after a time window. The process of dropping, keeping and assessing link(s) are similar to the smart-HOP mechanism.

Upon detecting a good link quality (from the average RSSI level in the DIO reply message), the MN continues the Data Transmission Phase and by observing ARSSI degradation, the MN starts the Discovery Phase. 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.

In the Discovery Phase, the MN broadcasts a burst of DIS control messages, receiving replies from all neighbor APs in a non-conflicting basis. The ARSSI level is embedded in the unicast DIO reply and for each DIO reply received, the MN compares the ARSSI value with T_h. If it is unsatisfactory (i.e., ARSSI below T_h), the MN continues broadcasting DIS bursts periodically. Upon detecting a high quality link (i.e., ARSSI above T_h), the Discovery Phase stops and the MN resumes regular data communication (with the new preferred parent) – Data Transmission Phase.

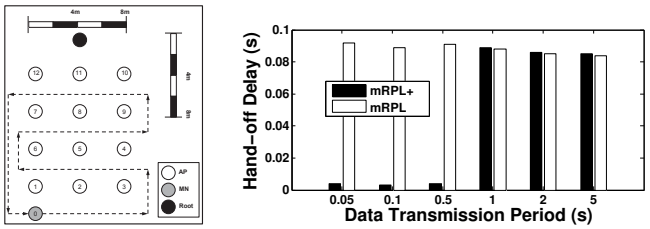
mRPL+. mRPL+ is a combination of the hard and soft handoff approaches within the RPL routing. The main contribution of mPL+ the introduction of overhearing mechanism, which enables continuous cooperation between the MN and neighbor APs in a single radio network. The APs overhear the link (between MN 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. By enabling the promiscuous mode, the MAC sub-layer accepts all frames received from the physical layer. The sniffer node receives all packets in its vicinity and stores the useful information in a table. mRPL+ defines some rules for detecting a physical mobility, which are described as below:

- The link quality degradation, which is observed by enabling the MAC layer acknowledgement and measuring the ETX values identify node mobility.
- Expiration of parent unreachability timer upon detecting inactivity in the link reveals node mobility. This event is followed by a unicast DIS transmission to the current AP. If the ETX level is unsatisfactory, handoff starts;
- Getting a message from serving AP that notifies the low link quality situation shows node mobility.

The soft handoff runs as a default and the hard handoff is called in three conditions:

- Receiving no response from the serving AP within a time frame (e.g. 100 ms),
- Receiving no response from neighbor APs before disconnecting from the serving AP (e.g. 500 ms before handoff mechanism starts),
- Observing low traffic network, considering the packet exchange intervals (e.g. $>$ 100 ms).

By enabling the overhearing mechanism, the neighbor APs overhear the communication between the MN and the serving AP. Unlike mRPL with hard handoff model, the neighbor APs are proactive by probing the MN. The potential APs with good quality links with the MN sends a beacon to inform their existence. In this model, the current AP replies if and only



(a) (b)

Fig. 1. (a) nodes' deployment, and (b) handoff delay of mRPL and mRPL+ with different data transmission periods [7].

if the ARSSI of a number of consecutive packets received from the MN is less than a predefined threshold (T_ℓ). In the meanwhile, the neighbor APs overhear links with the MN. After each data transmission between the MN and its serving AP, neighbor APs measure the RSSI. By getting an ARSSI above a certain threshold (T_h), the neighbor AP sends a DIO to the MN, holding the ARSSI measurement and the address of the AP (ID or IP). If the MN detects mobility based on one of the above rules (e.g. getting a message with $ARSSI < T_\ell$ from the current AP, say AP_1), and receive a message from a neighbor AP with $ARSSI > T_h$, say AP_2, then the MN switches quickly from AP_1 to AP_2.

Consider the case where the MN detects mobility in the network (in this example, ARSSI of AP_2 goes below T_ℓ) and it has no message received from the neighbor APs concerning their willingness for future service, then the MN resumes a hard handoff. In this process, the MN broadcasts bursts of DIS messages to neighbor APs to get replies from high quality links. This phase continues until finding a proper option for future data transmission. The MN simultaneously delivers data to the last preferred AP to keep the opportunity of delivering some of the messages. The possibility of successful data transmission during the Discovery Phase is low, because MN drops those packets that are generated during burst of DIS transmission. Moreover, the Discovery Phase has been planned to be very short (less than 100 ms), which eliminates data packet losses in applications with data generation rate less than 100 ms. It is rare to collect data with such a high rate in application like health monitoring.

IV. Problem description

As described previously in Section III, we have designed and implemented several handoff mechanisms for conventional IoT architectures (smart-HOP, mRPL and mRPL+). Experimental results indicated that mRPL and mRPL+ are able to provide network connectivity and responsiveness by switching between parents in a short period of time while successfully delivering most of data packets.

Figure 1 depicts the handoff delay of mRPL and mRPL+ for the case of having a mobile node (MN) and a number of Access Points (APs), while MN is moving in the deployment area, and switching from one AP to another. The result shows that the soft handoff approach leads to delays of approximately 4 ms in low data transmission periods (0.05, 0.1 and 0.5 sec). The soft handoff mechanism is unable to provide short delays in higher data periods as the underlying algorithms depend on the information that comes from the data exchanges between the MN and APs.

The preliminary result in Figure 1(b) illustrates the case where the mobility solution in a conventional network architecture fails even with the soft handoff approach. To further elaborate the problem, and to address the need to design novel algorithms and technologies for mobility management, two simple examples are presented next. Figures 2(a) and 2(b) show the situation where two mobile nodes are moving either in one direction or in opposite direction. Note that the number of mobile nodes in a real application may be much higher. It is highly likely that these mobile nodes select the same APs based on their link quality values. Our solution is to integrate an SDN controller within the IoT network as depicted in Figures 2(c) and 2(d), and thus make better handoff decisions and eventually better network scheduling for data transmission.

Furthermore, we have previously designed standalone SDN controller for IoT networks that communicates to Contiki nodes [23] through a TCP connection. In this work, we were envisioning the design of an SDN controller within the Contiki OS in order to enable employing IoT technologies within the SDN controller. Many challenges are involved in the new design, consisting of modeling OpenFlow components and OpenFlow messages.

V. Mobility management framework

In this work, we outline the design of a mobility management mechanism to support seamless 0-delay handoffs. It also ensures that the data packets reach to the destination through multiple paths. Our approach can be applied to various protocol designs within the IoT domain, such as ZigBee, RPL, and TSCH, as well as different SDN controller technologies, such as OpenFlow [19].

Seamless handoff. The type of handoff is dictated by the capabilities of the radio and the network architecture. 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. The process of disconnection and reconnection may imply short delays, which in turn will increase packet losses. Moreover, soft handoff is doable in networks with either multiple radios or with high traffic [7]. In new networking architecture that complies with the Fog computing architecture, a local controller node, known as SDN, is aware of the network situation. In a seamless handoff mechanism, the MN keeps sending its data to a number of APs, while one of them is the current AP (parent node). The MN switches instantly from one AP to another when the link quality degrades. This process has no delay as the decision has been made beforehand, and there is no packet losses as MN keeps multicasting data to neighbor nodes.

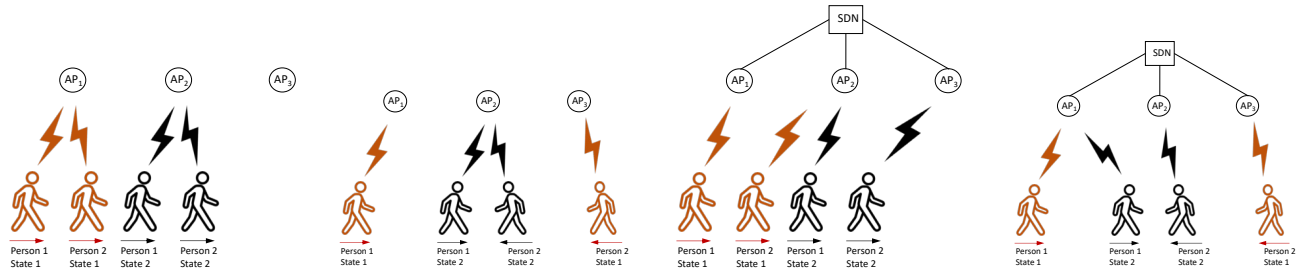


Fig. 2. (a) two people moving in same direction, and choosing same APs, (b) two people moving in opposite directions, and choosing same APs, (c) and (d) SDN-based solution that prevents inaccurate AP selection.

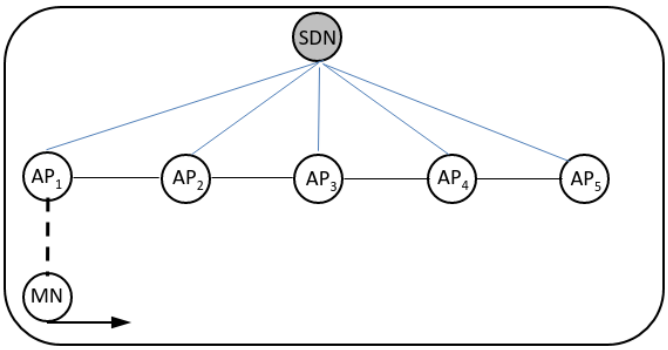


Fig. 3. A Fog computing IoT architecture with an SDN controller, a set of APs and a mobile node.

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 3 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 4 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. The current multipath approach follows an existing opportunistic routing, where the first parent node that successfully receives a packet from the MN, it forwards to the destination, and the other parents keep silent. 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 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 3. 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 and above the higher threshold 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.

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

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

Where $Q(.)$ is the complementary distribution function of the standard Gaussian, i.e., $Q(x) = \int_x^\infty (1/\sqrt{2\pi}) e^{-t^2/2} dt$, $R_a(i)$

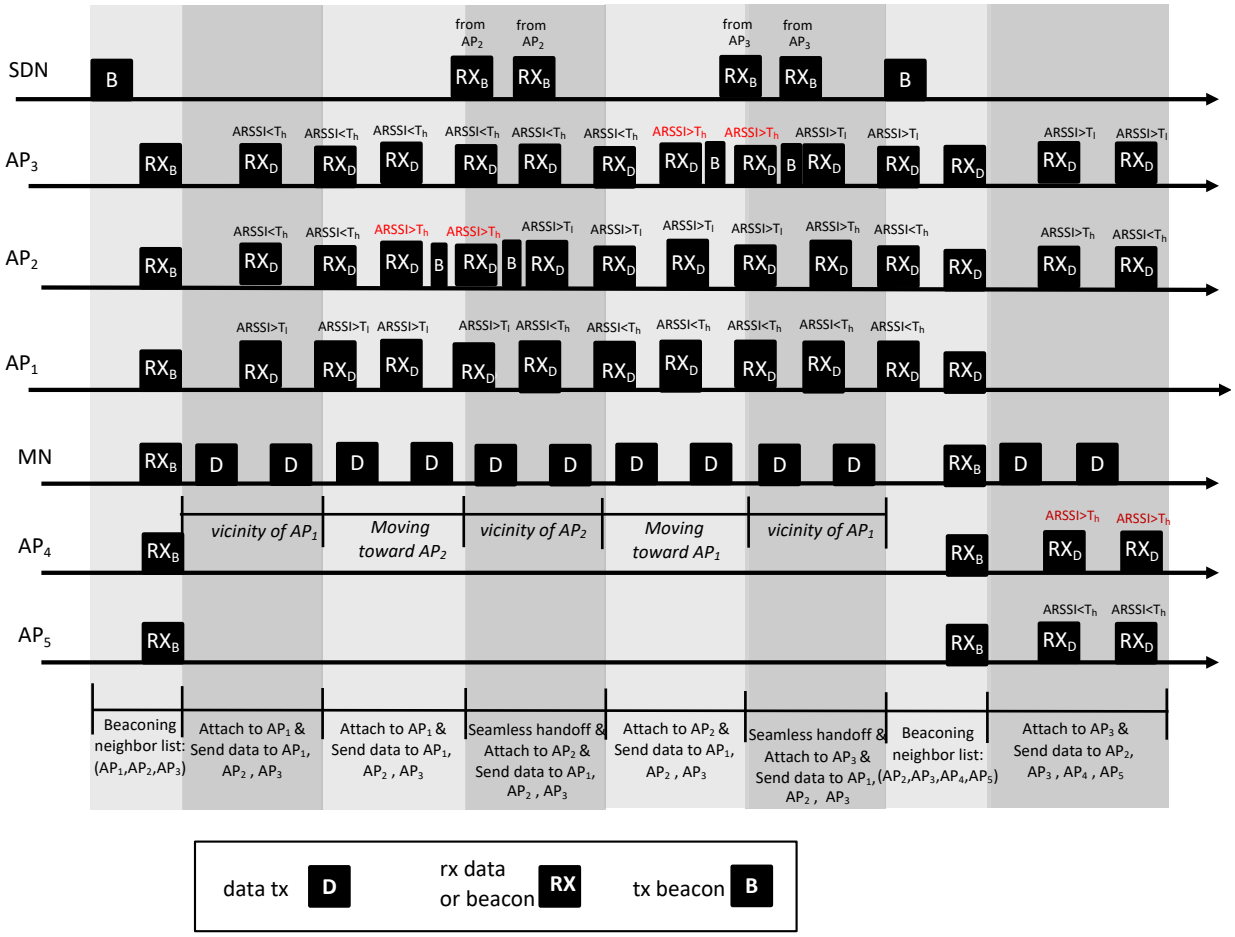


Fig. 4. As the MN moves across the horizontal path, the corresponding ARSSI changes over time and there is no need for any beaconing in the seamless handoff but if the parent that is the MN is going to attach to, does not exist in the MN's parent list a beacon from the SDN is required.

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 [34]:

$$R(i) = P_t - \overline{PL(d_0)} - 10\eta log_{10}(i/d_0) - X_\sigma \quad (1)$$

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).

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

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 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.

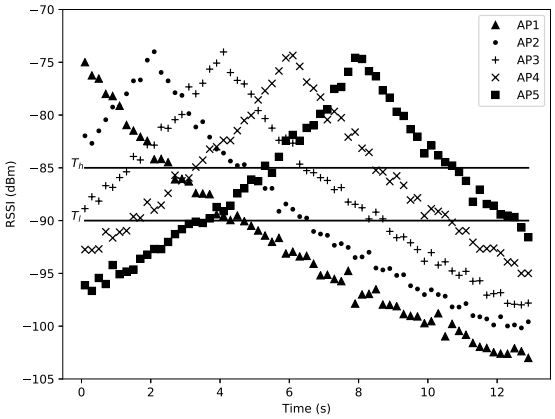


Fig. 5. The RSSI measured at the APs when the MN moves with a constant speed of 1m/s and the APs are statically placed at specified locations, as in Figure 3.

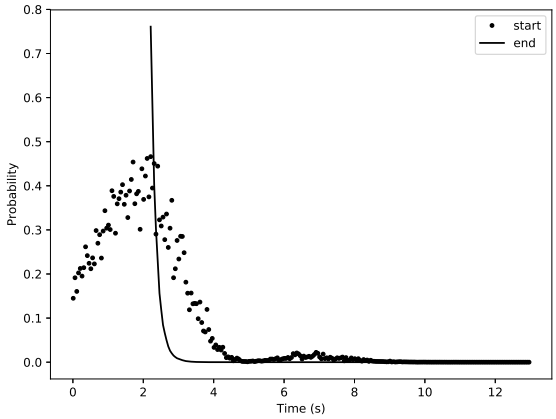


Fig. 6. Probability of starting handoff as indicated by dots, gets to its maximum after 2.2 seconds. Assuming that seamless handoff starts at that specific time slot, the conditional probability of ending the handoff as shown by the line, is high at exactly the same time instance.

$$P_{\text{Seamless}}(E(e) \mid S(s)) =$$

$$\begin{cases} P(R_b(e) \geq T_h) & e = s \\ \left[\prod_{i=s}^{e-1} P(R_b(i) \leq T_h)\right] & \\ \times P(R_b(e) \geq T_h), & e \geq s \end{cases} \tag{3}$$

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.

Handoff delay is the duration that takes from the starting slot to the ending slot. The main achievement of a seamless handoff mechanism is the zero delay, meaning that $e = s$. Based on Equation 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.

$$\text{Delay}(s,e) = \sum_{s=1}^{k-1} \sum_{e=s+1}^{k} ((e-s) \times P(E(e) \mid S(s)) \times P(S(s))) \tag{4}$$

According to this model, if we assume some realistic values for the parameters, we can have a better understanding of the performance of the handoff mechanism. 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. So as the MN moves in a horizontal path in Figure 3 the value for ARSSI in different APs changes as depicted in Figure 5. 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 which are located at position 0 and 4 in the X axis respectively, one can calculate the probability of starting the seamless handoff and the conditional probability of ending the handoff as shown in Figure 6. 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, using the proposed framework and taking advantage of the parameters that SDN provides for the mobile node, makes it possible to reduce the handoff delay to zero in seamless handoff mechanism compared to average 4ms in soft handoff in mRPL+ and 10ms in hard handoff.

VI. 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 a Software-Defined Network (SDN) controller. We summarise our previous hand-off approaches (smartHOP,

mRPL and mRPL+) and elaborate on the MobiFog seamless handoff mechanism. While our approach can be instantiated in different IoT protocols (e.g. ZigBee, RPL, TSCH), in this paper we use the RPL protocol. 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.

ACKNOWLEDGMENT

This work was supported by the Swedish Foundation for Strategic Research via the FiC project, and by the Swedish Research Council (Vetenskapsrådet), through the MobiFog starting grant, and by the Swedish Knowledge Foundation (KKS) through the FlexiHealth Prospekt, and the EU Celtic_Plus/Vinnova project, Health5G (Future eHealth powered by 5G).

REFERENCES

[1] 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*, vol. 24, no. 3, pp. 10–16, 2017.

[2] S.-M. Chun and J.-T. Park, "A mechanism for reliable mobility management for internet of things using coap," *Sensors*, vol. 17, no. 1, p. 136, 2017.

[3] A. V. Dastjerdi and R. Buyya, "Fog computing: Helping the internet of things realize its potential," *Computer*, vol. 49, no. 8, pp. 112–116, 2016.

[4] T. Chen, M. Matinmikko, X. Chen, X. Zhou, and P. Ahokangas, "Software defined mobile networks: concept, survey, and research directions," *IEEE Communications Magazine*, vol. 53, no. 11, pp. 126–133, 2015.

[5] 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*. Springer, 2012, pp. 131–146.

[6] H. Fotouhi, D. Moreira, and M. Alves, "mrpl: Boosting mobility in the internet of things," *Ad Hoc Networks*, vol. 26, pp. 17–35, 2015.

[7] H. Fotouhi, D. Moreira, M. Alves, and P. M. Yomsi, "mrpl+: A mobility management framework in rpl/6lowpan," *Computer Communications*, vol. 104, pp. 34–54, 2017.

[8] H. Fotouhi, M. Alves, M. Z. Zamalloa, and A. Koubâa, "Reliable and fast hand-offs in low-power wireless networks," *IEEE Transactions on Mobile Computing*, vol. 13, no. 11, pp. 2620–2633, 2014.

[9] M. Ha, S. H. Kim, and D. Kim, "Intra-mario: A fast mobility management protocol for 6lowpan," *IEEE Transactions on Mobile Computing*, vol. 16, no. 1, pp. 172–184, 2016.

[10] G. Bag, M. Raza, K.-H. Kim, and S.-W. Yoo, "Lowmob: Intra-pan mobility support schemes for 6lowpan," *Sensors*, vol. 9, no. 7, pp. 5844–5877, 2009.

[11] J. Ko and M. Chang, "Momoro: Providing mobility support for low-power wireless applications," *IEEE Systems Journal*, vol. 9, no. 2, pp. 585–594, 2014.

[12] 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)*. IEEE, 2014, pp. 200–209.

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

[14] R. Mahmud, R. Kotagiri, and R. Buyya, "Fog computing: A taxonomy, survey and future directions," in *Internet of everything*. Springer, 2018, pp. 103–130.

[15] B. Tang, Z. Chen, G. Hefferman, T. Wei, H. He, and Q. Yang, "A hierarchical distributed fog computing architecture for big data analysis in smart cities," in *Proceedings of the ASE BigData & SocialInformatics 2015*. ACM, 2015, p. 28.

[16] M. Chiang, S. Ha, I. Chih-Lin, F. Risso, and T. Zhang, "Clarifying fog computing and networking: 10 questions and answers," *IEEE Communications Magazine*, vol. 55, no. 4, pp. 18–20, 2017.

[17] N. McKeown, "Software-defined networking," *INFOCOM keynote talk*, vol. 17, no. 2, pp. 30–32, 2009.

[18] D. Kreutz, F. Ramos, P. Verissimo, C. E. Rothenberg, S. Azodolmolky, and S. Uhlig, "Software-defined networking: A comprehensive survey," *arXiv preprint arXiv:1406.0440*, 2014.

[19] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner, "Openflow: enabling innovation in campus networks," *ACM SIGCOMM Computer Communication Review*, vol. 38, no. 2, pp. 69–74, 2008.

[20] T. Luo, H.-P. Tan, and T. Q. Quek, "Sensor openflow: Enabling software-defined wireless sensor networks," *IEEE Communications letters*, vol. 16, no. 11, pp. 1896–1899, 2012.

[21] 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)*. IEEE, 2014, pp. 71–75.

[22] 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)*. IEEE, 2015, pp. 513–521.

[23] 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)*. IEEE, 2016, pp. 1–6.

[24] 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.

[25] B. Afzal, S. A. Alvi, G. A. Shah, and W. Mahmood, "Energy efficient context aware traffic scheduling for iot applications," *Ad Hoc Networks*, vol. 62, pp. 101–115, 2017.

[26] S. Rekik, N. Baccour, M. Jmaiel, K. Drira, and L. A. Grieco, "Autonomous and traffic-aware scheduling for tsch networks," *Computer Networks*, vol. 135, pp. 201–212, 2018.

[27] D. Dujovne, T. Watteyne, X. Vilajosana, and P. Thubert, "6TiSCH: deterministic IP-enabled industrial internet (of things)," *IEEE Communications Magazine*, vol. 52, no. 12, pp. 36–41, 2014.

[28] "802.15.4-2015: IEEE standard for local and metropolitan area networks–part 15.4: Low-rate wireless personal area networks (LR-WPANs) amendment 1," *IEEE Std.*, 2015.

[29] M. R. Palattella, N. Accettura, M. Dohler, L. A. Grieco, and G. Boggia, "Traffic aware scheduling algorithm for reliable low-power multi-hop ieee 802.15. 4e networks," in *2012 IEEE 23rd International Symposium on Personal, Indoor and Mobile Radio Communications-(PIMRC)*. IEEE, 2012, pp. 327–332.

[30] R. Soua, P. Minet, and E. Livolant, "Modesa: an optimized multichannel slot assignment for raw data convergecast in wireless sensor networks," in *2012 IEEE 31st international performance computing and communications conference (IPCCC)*. IEEE, 2012, pp. 91–100.

[31] R. Soua, E. Livolant, and P. Minet, "Musika: A multichannel multi-sink data gathering algorithm in wireless sensor networks," in *2013 9th International Wireless Communications and Mobile Computing Conference (IWCMC)*. IEEE, 2013, pp. 1370–1375.

[32] N. Accettura, M. R. Palattella, G. Boggia, L. A. Grieco, and M. Dohler, "Decentralized traffic aware scheduling for multi-hop low power lossy networks in the internet of things," in *2013 IEEE 14th International Symposium on" A World of Wireless, Mobile and Multimedia Networks"(WoWMoM)*. IEEE, 2013, pp. 1–6.

[33] R. Soua, P. Minet, and E. Livolant, "Wave: a distributed scheduling algorithm for convergecast in ieee 802.15. 4e tsch networks," *Transactions on Emerging Telecommunications Technologies*, vol. 27, no. 4, pp. 557–575, 2016.

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