Accepted in IEEE GLOBECOM - 2016, For personal use only

Mobility-Aware Flow-Table Implementation in Software-Defined IoT

Samaresh Bera and Sudip Misra
Department of Computer Science and Engineering
Indian Institute of Technology, Kharagpur, 721302, India
Email: s.bera.1989@ieee.org, smisra@sit.iitkgp.ernet.in

Mohammad S. Obaidat, *Fellow of IEEE*, *Fellow of SCS*
Department of Computer and Information Science
Fordham University, New York, 10458, USA
Email: msobaidat@gmail.com

***Abstract*—In this paper, we propose a mobility-aware flow-table implementation scheme with an aim to maximize overall network performance in software-defined IoT. The proposed scheme consists of two components — *path estimator* and *flow-manager*. The path estimator predicts future locations of end devices present in the network, and delivers info to the flow-manager. Based on predicted locations, the flow-manager implements forwarding rules at access devices (ADs) in the network, so that adequate actions for incoming requests can be taken immediately without asking the controller. We use order-k Markov predictor to predict the next possible locations of the end devices. We consider a practical scenario of an IoT environment, in which both static and mobile devices are present. Extensive simulation results show that the proposed scheme is beneficial for improving network performance in terms of energy consumption and message overhead for flow-table implementation, while predicting the future locations of the devices. We show that the proposed scheme is capable of enhancing the overall network performance approximately by 50%.**

***Index Terms*—Software-Defined Networking, Internet of Things, Wireless Access Network, Flow-Table, Mobility, Markov Predictor**

I. Introduction

Due to the advent features of software-defined networking (SDN) [1], it is getting interests among the researchers to support real-time application-specific requirements. In SDN, network-specific control strategies are defined by a centralized controller, while decoupling the *control-plane* from the forwarding devices, known as *data-plane*. Concurrently, internet of things (IoT) is an emerging technology to digitize everything for the betterment of connected world [2]. Consequently, main backbone of the IoT is to connect every network devices together and to control them in a unified manner. Thus, SDN-based solution approaches are one of the most feasible solutions to meet such requirements, while leveraging global view of the network.

Typically, an IoT environment consists of both stationery and mobile devices, which monitor different parameters in the environment, and communicate with access devices (ADs) to exchange their real-time information. Therefore, a flow-table is maintained at each of the ADs in order to take adequate actions for incoming requests from end devices[1]. In such a scenario, the flow-table rules need to be optimally managed at the ADs, depending on the presence of end devices and their requests. However, existing SDN-based solution approaches for flow-table implementation either considered the static behavior of the network or mainly focused on backbone networks, where dynamic behavior of the network is very low. Consequently, there is a need to have an optimal flow-table implementation strategy in software-defined IoT (SDIoT) networks for efficient network management, while considering users' quality-of-experience (QoE). To address such issues, two solution approaches are feasible to update the flow-table for information forwarding — a) reactive – ADs inform controller after receiving requests, and the controller defines forwarding rules; b) proactive – controller defines forwarding rules proactively based on the end device's mobility patterns, and instructs the ADs to update their flow-table rules.

In this paper, we propose a mobility-aware flow-table implementation scheme for software-defined IoT environments with an aim to maximize overall network performance, while considering end users' movement in the network. The proposed scheme consists of two components — path estimator and flow-manager. The path estimator uses order-k Markov predictor [3], [4] to predict the future locations of end devices based on the past history of visited locations and time of arrivals. Further, based on the predicted future location, the flow-manager defines the forwarding rules. Finally, the ADs adapt the forwarding rules defined by the flow-manager to take adequate actions for incoming requests from end devices. Extensive simulation results show that the proposed scheme is beneficial to update the forwarding rules in an adaptive manner, while minimizing energy consumption and message overhead in the network. In brief, the contribution in this paper are as follows:

- We propose a mobility-aware flow-table implementation scheme to maximize QoE of users, while minimizing energy consumption and message overhead in the network. The problem is challenging because of the presence of heterogeneous devices and capacity constraints of the network devices.
- The proposed scheme consists of two components — path estimator and flow-manager. The path estimator predicts future locations of users and flow-manager implements

[1]In this paper, the term 'user' and 'end device' are used to denote the same component.

forwarding rule at ADs. The order-k Markov predictor is used to predict the future locations.

- Simulation results show that the proposed scheme is beneficial for improving the network performance, while considering movement of the users in the network.

The rest of the paper is organized as follows. Section II discusses existing works in the context of forwarding rule placement in SDN. Section III presents overall system architecture. We present the mobility-aware flow-table implementation scheme in Section IV. Section V shows the performance of the proposed scheme. Finally, we conclude the paper in Section VI, while presenting some future research directions.

II. Related Work

There are several existing works in the literature which focus on forwarding rule management at network switches [5]–[13]. Li et al. [5] proposed an optimal rule placement scheme for the backbone network switches from the aspects of SDN. The authors addressed the issues with capacity constraints of the ternary content-addressable memory (TCAM) available at switches. The rules are replaced with a new one depending on the traffic pattern and devices' availability. On the other hand, Giroire et al. [6] proposed an energy-aware routing scheme using SDN technology. In such a scheme, unused links in the network are put into sleep mode to save energy, while considering quality-of-service (QoS) of the network. Similarly, Markiewicz et al. [7] proposed an optimal energy consumption technique for SDN-enabled network, while considering dynamic traffic in the network.

Vawter et al. [8] proposed an optimal traffic management policies to minimize unwanted traffic in the network, while maximizing network performance. The authors developed a test-bed to analyze the network performance of such SDN-enabled network. Huang et al. [9] proposed a joint optimization approach for optimal rule placement and traffic engineering in the network. Heuristic algorithm is used to optimized the network performance. Similarly, in [10], the authors proposed an optimal rule partition and allocation scheme in the backbone network switches. The rules at the switches are handled in an efficient manner, depending on the status of the network. Therefore, the rules are managed in a proactive manner to minimize network delay and buffers at the switches. Li et al. [11] proposed a rule placement strategy to deal with predictable and unpredictable flows in the network, which is different from the traditional packet-driven rule caching approaches.

Zhang et al. [12] proposed an integer linear programming (ILP)-based scheme to optimize rule placement policies at network switches, while considering given firewall policy and capacity constraints of the switches. On the other hand, Ma et al. [13] proposed a network function virtualization (NFV) scheme, depending on the dynamic requirements of the network. In such a scheme, issues related to traffic-aware middleboxes placement in the network are addressed.

However, detailed analysis of the existing works reveals that there is a research lacuna on rule placement policies in the context of IoT, in which both static and mobile devices are present. Due to the presence of mobile devices in the network, the ADs need to frequently update the forwarding rules for incoming requests to take adequate actions. However, the existing solution approaches did not consider the dynamic behavior of the network in the presence of mobile devices in an SDIoT environment. Therefore, in this paper, we propose a mobility-aware forwarding rule management policies at the access devices in SDN-enabled network.

III. System Model

In this Section, we present the proposed architecture and problem statement for flow-table implementation in an SDIoT. Figure 1 presents a schematic view of an IoT environment enabled with SDN. We consider that the IoT environment consists of heterogeneous devices (such as sensors, mobile devices, and peripheral devices), which communicate with access points (APs) and base stations (BSs) to exchange real-time information. Additionally, we also consider that the end devices can be both stationery and mobile in nature, as considered in IoT. The AP and BS forward data traffic based on forwarding rules decided by a centralized SDN controller, as depicted in Figure 1. Therefore, the flow-table rules are dynamically updated by the controller, depending on application-specific requirements of IoT users.

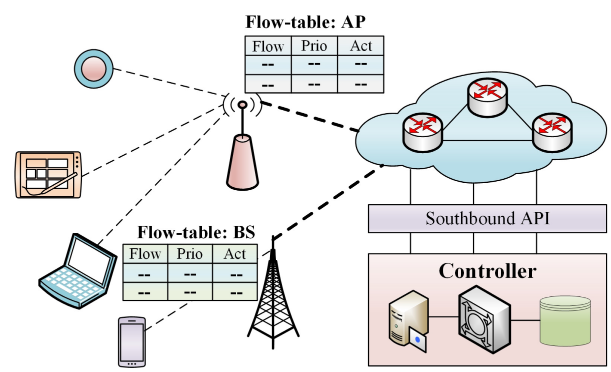

Fig. 1: Proposed architecture of software-defined IoT

The data traffic accessed by the AP and the BS are further forwarded through the backhole communication network (i.e., routers and switches). Based on the received data, the controller takes adequate decisions for implementing forwarding rules at the backhole communication networks and access networks. Consequently, the AP and BS adapt the flow-table rules and take adequate actions for an incoming data traffic from the end devices. For simplicity, we do not focus on the forwarding issues present in the backbone networks.

A. Problem Statement

As shown in Figure 1, forwarding rules at the AP and the BS are dynamically changed, depending on users' positions and requirements. Due to the resource constraint nature of the access devices (ADs), limited forwarding rules can be entered in the flow-tables associated with the devices [14].

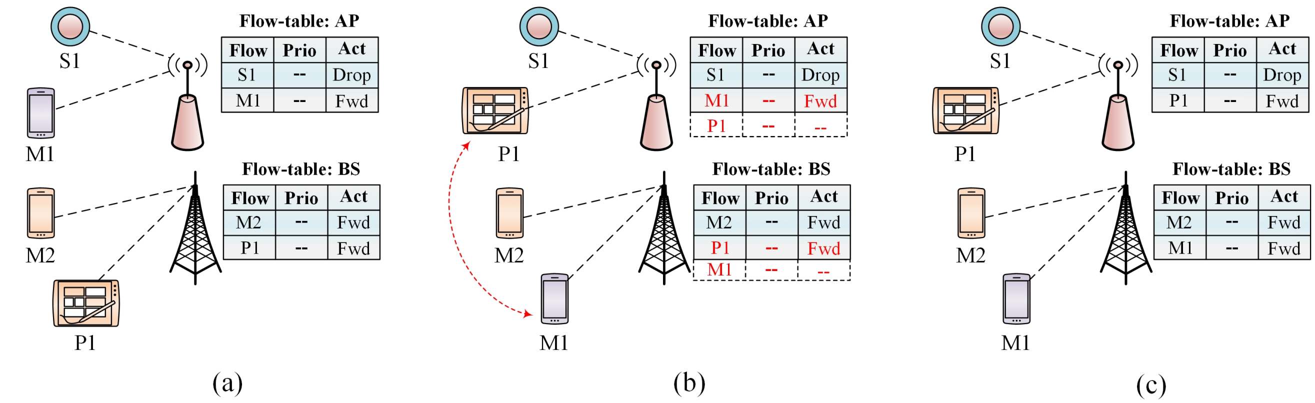

Fig. 2: Illustrative example: (a) Scenario 1: ADs are capable of handling requests from all devices associated to them; (b) Scenario 2: ADs are incapable of handling requests due to capacity constraint, in the presence of mobile devices; (c) Scenario 3: Adjusted flow-table with rules, which is capable of handling requests from all devices

Let consider a wireless network comprises of multiple ADs (combination of BSs and APs), which is denoted by the set $\mathcal{A} = \{A_1, A_2, \ldots, A_n\}$, where $n \in \mathcal{N}$. Let also consider that maximum R number of rules can be entered in each AD due to the capacity constraints. Mathematically,

$$\sum_{i=1}^{r} A_j(\mathcal{R}_i) \leq R \quad j \in \mathcal{N} \tag{1}$$

On receiving a new request, two solutions are feasible: a) the request is discarded; b) the request is entered, while removing an old entry, while the capacity is full.

Let also consider that $d \in \mathcal{N}$ number of end devices are present in the network, which is denoted by the set $\mathcal{D} = \{D_1, D_2, \ldots, D_d\}$. Each device may have multiple rules associated with it, depending on different requests. Therefore, total rule-space handled by the ADs in the network cannot be more than the total capacity. Mathematically,

$$\sum_{j=1}^{d}\sum_{k=1}^{r} D_{j,k} \leq \sum_{i=1}^{n}\sum_{l=1}^{r} \mathcal{R}_{i,l} \tag{2}$$

where $D_{j,k}$ denotes number of rule-space of j^{th} device. The number of rule-space handled by i^{th} AD is denoted as $\mathcal{R}_{i,l}$.

We consider an accuracy factor $(\mathcal{F}_i)$ for an AD $i \in \mathcal{A}$, which is denoted as follows:

$$\mathcal{F}_i = \frac{\mathcal{R}_{i,present}}{\mathcal{R}_{i,total}} \tag{3}$$

where $\mathcal{R}_{i,present}$ denotes the number of different forwarding rules present at i^{th} AD, and $\mathcal{R}_{i,total}$ denotes required number of forwarding rules at the AD to handle all requests. The objective is to maximize the accuracy for all ADs in the network in order to improve overall network performance. Mathematically,

$$\begin{aligned} \text{maximize} \quad & \sum_{i=1}^{n} \mathcal{F}_i \\ \text{subject to} \quad & 0 \leq \mathcal{F}_i \leq 1 \end{aligned} \tag{4}$$

Problem: The forwarding rule is required to be updated adequately in order to improve the network performance. Consequently, the ADs adapt the rules associated with an end device defined by the controller in its table. However, the ADs may not have adequate forwarding rule information due to the mobile nature of the end devices. Additionally, rule for a particular request may be replicated in the network. Moreover, due to the resource constraint nature of the ADs, new requests may not be served, while the rule-space capacity is full. Therefore, it is required to have an adaptive flow-table implementation scheme, which manages the forwarding rules adequately to maximize the accuracy $(\mathcal{F}_i)$ for an AD $\in \mathcal{A}$.

B. Illustrative Example

Figure 2 presents an illustrative example consisting of three different scenarios. We consider a network consisting of an AP and a BS, and few heterogeneous devices such as sensor, smart phones, and PDA. Depending on the positions and requirements, forwarding rules are defined at the AP and the BS, as shown in Figure 2. We consider that both the AP and the BS has limited rule-space capacity.

In scenario 1, the flow-table at the AP consists of rules associated with S1 and M1, and the BS consists of rules associated with M2 and P1. Therefore, the rule-space at both the ADs are well-utilized, and there is no problem to take adequate actions for incoming requests from end devices.

Let consider that M1 and P1 move to the vicinity of the BS and AP, respectively. Therefore, the rules are required to be modified. However, on receiving request from M1 and P1, the ADs cannot allocate the rule-space to serve the request as the capacity is full. So, we need to remove the existing rule(s) which is (are) no more required, and need to insert new rule(s) to serve the request.

Scenario 3 presents the modified flow-table rules, in which rules are added/removed based on end devices' positions and requirements. This can be done a reactive manner, i.e., after receiving request, rules can be added/removed. Consequently, network delay is increased, while considering the reactive process. To address this issue, we propose mobility-aware flow-table implementation scheme (which is proactive) to adequately implement the forwarding rules with an aim to maximize the network performance in an SDIoT environment.

IV. Mobility-Aware Flow-Table Implementation

The proposed framework is presented in Figure 3, and it is placed at the controller end. Consequently, the computational complexity is avoided at the ADs. The proposed model consists of different components — path estimator, flow manager, database, and flow-table — as presented in Figure 3. Path estimator predicts the future locations of end devices based on history data (refer to Section IV-A). Further, based on the predicted locations, the flow manager decides the flow-table rules (refer to Section IV-B), and the table is implemented on the associated ADs in order to provide seamless connectivity.

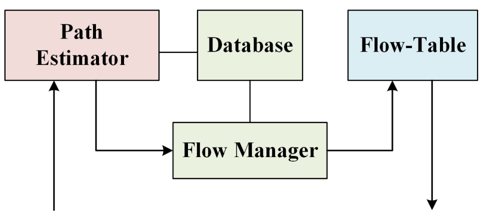

Fig. 3: Proposed framework for flow-table implementation

A. Path Estimation

We use order-k, $O(k)$, Markov predictor [3], [4] to estimate the future locations of end devices. The model consists of different tuples: $< \mathcal{S}, \mathcal{A}, P, R >$, where

- $\mathcal{S}$: Set of states of meaningful places visited by users (i.e., end devices), which is represented as $\mathcal{S} = \{s_1, s_2, \ldots, s_n\}$, $n \in \mathcal{N}$.
- $\mathcal{A}$: Set of actions taken on a particular state, i.e., time duration (d) before handoff[2] occurs. Therefore, $\mathcal{A}$ is represented as $\mathcal{A} = \{a_1, a_2, \ldots, a_n\}$, $n \in \mathcal{N}$.
- P: Probability of transition from one state to another state. Therefore, $P_{ij}(a_1)$ represents the probability of transition from state s_i to state s_j, $i \neq j$, when action a_1 is taken.
- R: Reward factor, which defines the prediction accuracy on whether handoff occurs after the time duration d.

For a given movement history set of a user, which is denoted as $\mathcal{H} = \{(s_1, t_1), (s_2, t_2), \ldots, (s_n, t_n)\}$, the path estimator estimates both time and location of next handoff. s denotes the state or location of the user, and t denotes the time of arrival to the state s. In order to predict the location and time of next handoff, we need to calculate the probability that a handoff will occur in the next Δt time period, while the current location and duration of stay at the location are given.

From $\mathcal{H}$, we extract the state history set $\mathcal{S} = \{s_1, s_2, \ldots, s_n\}$, and from $\mathcal{S}$ the order-k[3] location contexts $C = \mathcal{S}(n-k+1, n) = \{\mathcal{S}_{n-k+1}, \mathcal{S}_{n-k+2}, \mathcal{S}_n\}$. Now, we search for instances of the context c.

Therefore, for the given current context c, the path estimator predicts the duration of stay $\mathcal{D}_s$ at possible s locations which follow c. Mathematically,

$$\mathcal{D}_s = \{d_i | d_i = t_{i+1} - t_i, \text{ where } \mathcal{S}(i-k+1, i+1) = c_s\} \quad (5)$$

For each $\mathcal{D}_s$, we calculate the conditional probability $P_s(t \leq d < t + \Delta t)$ that the user will move to location s within Δt time after the current elapsed time t. Consequently, for given context c and elapsed time t, the probability of each user moving to each possible location s within Δt time is calculated as follow:

$$P(s|c,t) = P(s) P_s(t \leq d < t + \Delta t | c, t) \quad (6)$$

where $P(s)$ is the probability of every possible next location s, which can be calculated as follows:

$$P(s_{n+1} = a | \mathcal{H}) \approx \hat{P}(s_{n+1} = a | \mathcal{H}) = \frac{N(ca, \mathcal{H})}{N(c, \mathcal{H})} \quad (7)$$

where $N(ca, \mathcal{H})$ denotes the number of occurrence of ca in the history set $\mathcal{H}$. Accordingly, the Markov predictor predicts the most likely location s will be visited at $n+1$ time as follows:

$$s_{n+1} = \underset{a \in \mathcal{A}}{\operatorname{argmax}} (P(s_{n+1} = a)) \quad (8)$$

B. Flow-Table Implementation

As discussed in Section IV-A, the predicted next location is s_{n+1}, which is collected by the flow manager from the path estimator. The flow manager checks whether a handoff will occur for the next location s_{n+1} from current location s_n, which is denoted by an indicator variable as follows:

$$\mathcal{I} = \begin{cases} 1, & \text{if handoff occurs} \\ 0, & \text{otherwise} \end{cases} \quad (9)$$

For $\mathcal{I} = 1$, the flow manager inserts a new rule to the flow-table of an AD to be associated in the next time period, and deletes the rule from the currently associated AD. Consequently, the flow-table rules are managed in a reliable and efficient manner, so that accuracy factor $(\mathcal{F})$ is maximized. It is noteworthy that the objective is to predict the associated ADs for end devices instead of determining exact locations.

C. Proposed Algorithm

The algorithm for path estimation is presented in Algorithm 1. It is noteworthy that the presented algorithm is for one end device. However, the path estimator estimates the next locations for all end devices associated with different ADs in the network in a similar manner.

Algorithm 2 presents different steps followed by the flow manager to implement forwarding rules in the associated ADs. The steps are repeated for all ADs in the network.

[2]Handoff is considered as the change in associated ADs to an end device
[3]Current k ($k = 3$) instances are considered.

Algorithm 1: Algorithm for Path Estimator

Input: History set $\mathcal{H}$, current context c
Output: Next predicted location s_{n+1}

Extract the state of location history set $\mathcal{S}$ from $\mathcal{H}$;
Predict $\mathcal{D}_s$ at possible locations s according to Equation (5);
Calculate $P(s|c,t)$ according to Equation (6);
Predict next location s_{n+1} according to Equation (8);
Return s_{n+1};

Algorithm 2: Algorithm for Flow Manager

Input: Next location s_{n+1}, current location s_n
Output: Implement flow-table according to handoff

Check for handoff according to Equation (9) for location change from s_n to s_{n+1};
if $\mathcal{I} == 1$ **then**
    Decide new rule $\mathcal{R}$;
    Insert $\mathcal{R}$ to newly associated AD after Δt time;
    Delete $\mathcal{R}$ from currently associate AD;
else
    Do not change the rule;

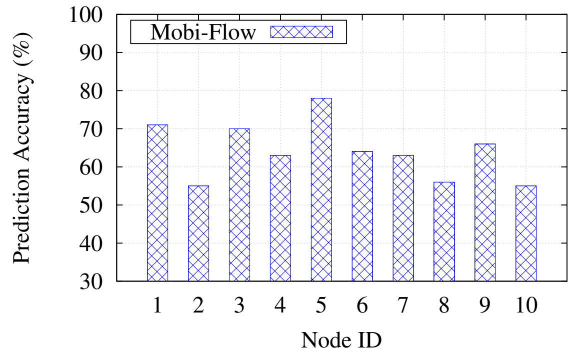

Fig. 4: Prediction accuracy for individual nodes

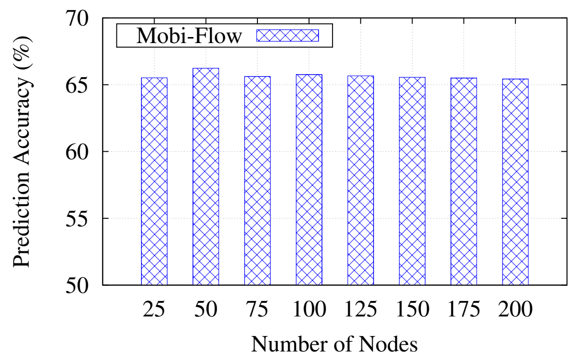

Fig. 5: Prediction accuracy for different number of nodes in the network

V. Performance Evaluation

A. Simulation Settings

We evaluate the proposed scheme in a discrete event simulator with different simulation parameters, as shown in Table I. We use the Gauss-Markov mobility model [15] for the mobile nodes in the network. Energy consumption for rule management is considered as the energy spent for transmit, receive, and computation at the ADs. Different performance metrics — prediction accuracy, message overhead, and energy consumption — are used to show the effectiveness of the proposed scheme. Henceforth, we use the term *Mobi-Flow* to denote the proposed scheme. On the other hand, the term 'conventional' is used to represent the existing scheme, where mobility of the end devices are not taken into account.

TABLE I: Simulation Parameters

Parameter	**Value**
Number of Nodes	200
Mobility Model	Gauss-Markov Mobility [15]
Number of ADs	25
Transmit Power [16]	2.2 W
Receive Power [16]	1.35 W

B. Results and Discussion

As discussed in Section IV, we evaluate the prediction accuracy of predicted locations for all nodes in the network. Figure 4 shows the prediction accuracy of associated ADs for individual nodes. We see that *Mobi-Flow* significantly predicts the associated ADs for individual nodes in the network. Similarly, Figure 5 presents the overall prediction accuracy in the network for different number of nodes in the network. In all cases, we get above 65% accuracy in the network. Intuitively, we can say that the customers' satisfaction factor also increases with the proposed scheme.

It is also important to measure message overhead in the network for rule-management at the ADs. We evaluate the message overhead as the number of messages exchanged between flow-manager and ADs for incoming requests from end-devices. Figure 6 presents the total message exchanged in the network for rule management at the ADs. We also see that the message overhead is minimized significantly, as the forwarding rules are placed at ADs well-before getting any new requests depending on the predicted locations. In contrast, in the 'conventional' one, ADs ask to the flow-manager for every new incoming requests from end-devices, which, in turn, maximizes the message overhead in the network for rule management. *Mobi-Flow* reduces the message overhead by approximately 45% over the conventional approaches.

Figure 7 depicts total energy consumption for rule management in the network. Energy consumption for rule management depends of the number of messages exchanged between flow-manager and ADs in the network. As the proposed scheme significantly reduces the message overhead in the network (as shown in Figure 6), energy consumption is

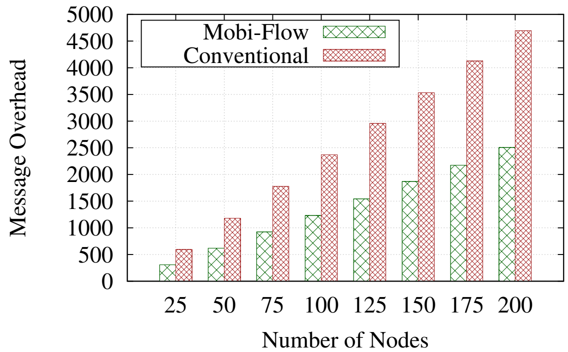

Fig. 6: Message overhead in the network for rule management

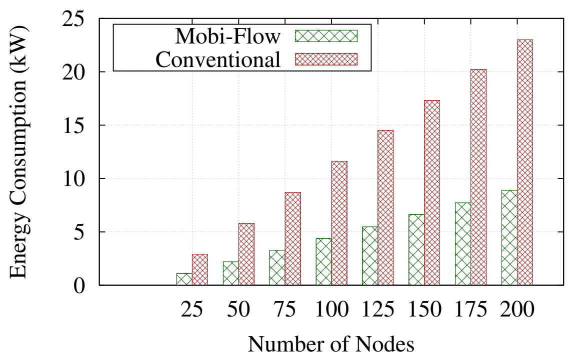

Fig. 7: Energy consumption for rule management

also minimized. We see that *Mobi-Flow* reduces the energy consumption by approximately 55% over the conventional schemes, in which devices' mobility is not considered.

VI. Conclusion

In this paper, we proposed a mobility-aware flow-table implementation scheme with an aim to maximize overall network performance of SDIoT. We used order-k Markov predictor-based scheme to predict the future locations of end devices in the network. Simulation results showed that the proposed scheme significantly reduces the message overhead and energy consumption for rule management approximately by 45% and 55%, respectively, while predicting the future locations of the devices in the network.

We plan to evaluate the proposed scheme in a real test-bed as the future extension of this work. In addition to this, future extension of this work also includes the maximization of prediction accuracy further.

References

[1] B. Nunes, M. Mendonca, X.-N. Nguyen, K. Obraczka, and T. Turletti, "A Survey of Software-Defined Networking: Past, Present, and Future of Programmable Networks," *IEEE Communications Surveys and Tutorials*, vol. 16, no. 3, pp. 1617–1634, 2014.

[2] L. Atzori, A. Iera, and G. Morabito, "The Internet of Things: A survey," *Computer Networks (Elsevier)*, vol. 54, no. 15, pp. 2787–2805, 2010.

[3] L. Song, U. Deshpande, U. C. Kozat, D. Kotz, and R. Jain, "Predictability of WLAN Mobility and Its Effects on Bandwidth Provisioning," in *Proc. of the IEEE INFOCOM*, Barcelona, Spain, Apr. 2006, pp. 1–13.

[4] Y. Chon, E. Talipov, H. Shin, and H. Cha, "SmartDC: Mobility Prediction-Based Adaptive Duty Cycling for Everyday Location Monitoring," *IEEE Transactions on Mobile Computing*, vol. 13, no. 3, pp. 512–525, Mar. 2014.

[5] H. Li, P. Li, and S. Guo, "MoRule: Optimized rule placement for mobile users in SDN-enabled access networks," in *Proc. of the IEEE GLOBECOM*, TX, Dec. 2014, pp. 4953–4958.

[6] F. Giroire, J. Moulierac, and T. K. Phan, "Optimizing rule placement in software-defined networks for energy-aware routing," in *Proc. of the IEEE GLOBECOM*, TX, Dec. 2014, pp. 2523–2529.

[7] A. Markiewicz, P. N. Tran, and A. Timm-Giel, "Energy consumption optimization for software defined networks considering dynamic traffic," in *Proc. of the IEEE International Conference on Cloud Networking (CloudNet)*, Luxembourg, Oct. 2014, pp. 155–160.

[8] I. Vawter, D. Pan, and W. Ma, "Emulation Performance Study of Traffic-Aware Policy Enforcement in Software Defined Networks," in *Proc. of the IEEE International Conf. on Mobile Ad Hoc and Sensor Systems*, Philadelphia, PA, Oct. 2014, pp. 775–780.

[9] H. Huang, S. Guo, P. Li, B. Ye, and I. Stojmenovic, "Joint Optimization of Rule Placement and Traffic Engineering for QoS Provisioning in Software Defined Network," *IEEE Transactions on Computers*, vol. 64, no. 12, pp. 3488–3499, Dec. 2015.

[10] J.-F. Huang, G.-Y. Chang, C.-F. Wang, and C.-H. Lin, "Heterogeneous Flow Table Distribution in Software-defined Networks," *IEEE Transactions on Emerging Topics in Computing*, 2015, DOI: 10.1109/TETC.2015.2457333.

[11] H. Li, S. Guo, C. Wu, and J. Li, "FDRC: Flow-driven rule caching optimization in software defined networking," in *Proc. of the IEEE ICC*, London, Jun. 2015, pp. 5777–5782.

[12] S. Zhang, F. Ivancic, C. Lumezanu, Y. Yuan, A. Gupta, and S. Malik, "An Adaptable Rule Placement for Software-Defined Networks," in *Proc. of the IEEE/IFIP International Conference on Dependable Systems and Networks*, Atlanta, GA, Jun. 2014, pp. 88–99.

[13] W. Ma, C. Medina, and D. Pan, "Traffic-Aware Placement of NFV Middleboxes," in *Proc. of the IEEE GLOBECOM*, San Diego, CA, Dec. 2015, pp. 1–6.

[14] C.-T. Chan, P.-C. Wang, S.-C. Hu, C.-L. Lee, and R.-C. Chen, *Information Networking: International Conference, ICOIN 2003.Revised Selected Papers*. Springer, 2003, ch. Scalable Packet Classification for IPv6 by Using Limited TCAMs, pp. 76–85.

[15] F. Bai and A. Helmy, *Wireless ad-hoc networks, Chapter 1: A survey of mobility models*. http://nile.usc.edu/helmy/important/Modified-Chapter1-5-30-04.pdf, 2004.

[16] M. Carvalho, C. Margi, K. Obraczka, and J. Garcia-Luna-Aceves, "Modeling energy consumption in single-hop IEEE 802.11 ad hoc networks," in *Proc. of the IEEE Intl. Conf. on Comp. Comm. and Net.*, Chicago, IL, 2004, pp. 367–372.