JOURNAL OF INFORMATION SCIENCE AND ENGINEERING 37, 839-858 (2021)
DOI: 10.6688/JISE.202107_37(4).0007

Management of Flow Table of SDN for Proactive Eviction Using Fuzzy Logic

GAN HUANG AND HEE YONG YOUN
Department of Electrical and Computer Engineering,
Sungkyunkwan University
Suwon, 16419 Korea
E-mail: {ahczhg; youn7147}@skku.edu

With the development of Software Defined Network (SDN), numerous researches have been conducted for improving the performance of SDN. In SDN flow table is used in OpenFlow switch for the routing of the packets. Due to the space limitation of flow table and switch capacity, various issues need to be resolved for effectively dealing with a large number of flows. The existing schemes typically employ a reactive approach such that evicted entries are decided only when timeout or table miss occurs. In this paper a novel proactive eviction scheme is proposed which employs hidden Markov model (HMM) to predict the probability of table miss of the entries. If the probability exceeds the preset threshold, fuzzy logic is used to select the entries for eviction considering match priority, idle time, and the number of unmatched flows via a new notion called eviction index. The proposed scheme is for efficient flow entry eviction before table miss actually occurs, which eventually increases the speed of flow management in SDN switch. Computer simulation reveals that the proposed scheme increases the match probability and prediction accuracy, and reduces the number of misses at least 10% compared to three existing entry eviction schemes.

Keywords: SDN, OpenFlow, flow entry eviction, HMM, fuzzy logic

1. INTRODUCTION

With the escalating scale of network infrastructure, more flexible and dynamic manipulation of the network is required using easily modifiable networking devices [1, 2]. Nowadays, the primary approach for satisfying such requirement is employing the software-defined networking (SDN) technology. As depicted in Fig. 1, a centralized control point is provided in SDN. Here the control plane perceives the network status and decides the flow paths, while the data plane is responsible for the transmission of packets [3]. The separation of the control layer from the data layer enables the programmability, increases functionality, and provides remote management between the components using a single open protocol [4]. The network and business applications are allowed to be effectively integrated with the aid of analytics, and the networking policies can be adjusted according to the changing condition and performance [5].

Using OpenFlow protocol, the controller controls the switch to add, update, and delete the entries of flow tables, reactively in response to packet arrival or proactively according to the status of the table [6]. When a flow enters the switch, its packet is matched with the entries of the flow table [7]. If an entry is matched, the instructions included in that entry is executed. Otherwise, table miss [8] occurs of which likelihood depends on the

Received July 3, 2019; revised August 20 & September 22, 2019; accepted November 5, 2019.
Communicated by Jiann-Liang Chen.

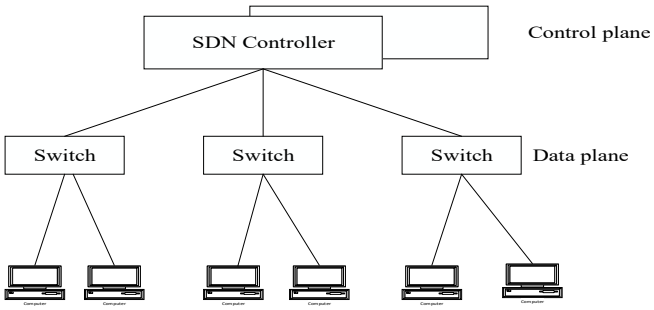


Fig. 1. The structure of SDN.

table configuration [9]. If the table-miss flow entries exist, the instructions included in that entry of the flow table specifies how to process the unmatched packet including dropping, passing to another table, or forwarding to the controller over the control channel via the packet-in message. The controller can delete or add the flow entries at any time. By default, the table-miss flow entries do not exist in the flow table. The configuration of a switch, for example adopting the OpenFlow Configuration Protocol, may override the default configuration and specify the different operation. Fig. 2 presents how a packet flow is handled inside a switch.

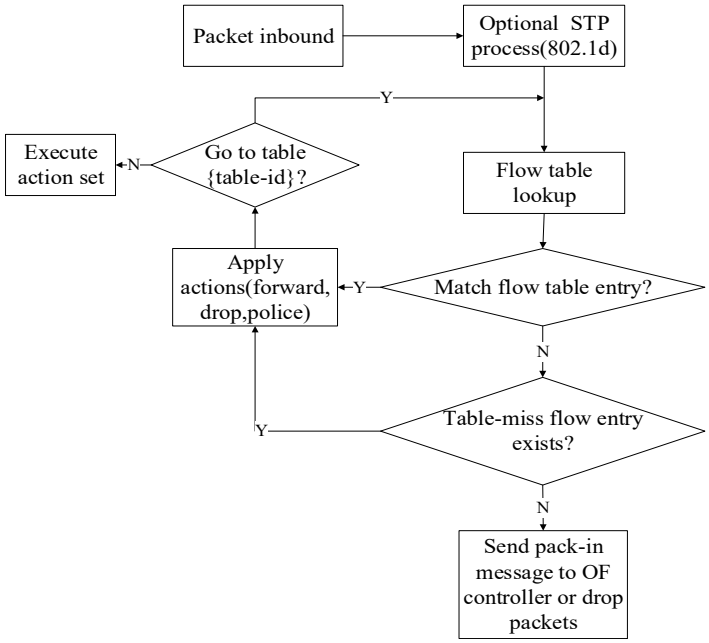


Fig. 2. Handling a packet flow in OpenFlow switch.

Due to the power, cost, and space constraint in OpenFlow switch, efficiently managing the flow entries is particularly important [10]. The service time of 80% of the flow

entries is usually less than 10 seconds while that of the other entries is about 100 milliseconds [11]. Various approaches have been proposed to improve the switch efficiency by freeing the space of flow entry, which is no longer needed [12-14]. Most of them are based on the adaptive table-miss prediction approach proposed to dynamically evict the flow entries and thereby maximize the flow table utilization [15]. They adopt a fixed or adaptive time-out scheme, or utilize the information on the usage of switch link, network dynamics, and so on. Also, special hardware component such as bloom filter is utilized to refer to the statistics of matches, and the transition frequency and probability between the hit state and wait state are also taken into account.

While there exist various dynamic flow entry eviction schemes, the switch efficiency is still low due to limited predictability of the usage of the entries. In this paper, thus, a new proactive flow entry eviction mechanism is proposed, adopting hidden Markov model (HMM) and fuzzy logic to accurately predict the likelihood of the usage of a flow entry. Here HMM is employed to predict the probability of the occurrence of table-miss of the entries. It maximizes the accuracy of prediction through the dynamic calculation of the likelihood of match. In addition, a new notion of 'eviction index' is adopted using the HMM in selecting the flow entry to be evicted of which match probability is lower than the threshold. In deciding the eviction index, fuzzy logic is used to analyze the match priority, idle time, and the number of unmatched flows in an integrated way. Finally, the flow entry of the highest eviction index is evicted. The performance of the proposed approach is evaluated by computer simulation based on Opendaylight and Matlab. It is also compared with the existing representative flow entry eviction schemes in terms of match probability and the number of misses. It indicates that the proposed scheme significantly increases the match probability and reduces table misses, at least about 10%. The existing issues with the flow table management and their solutions with the proposed scheme are depicted in Fig. 3, and the main contributions of the paper are summarized as follows.

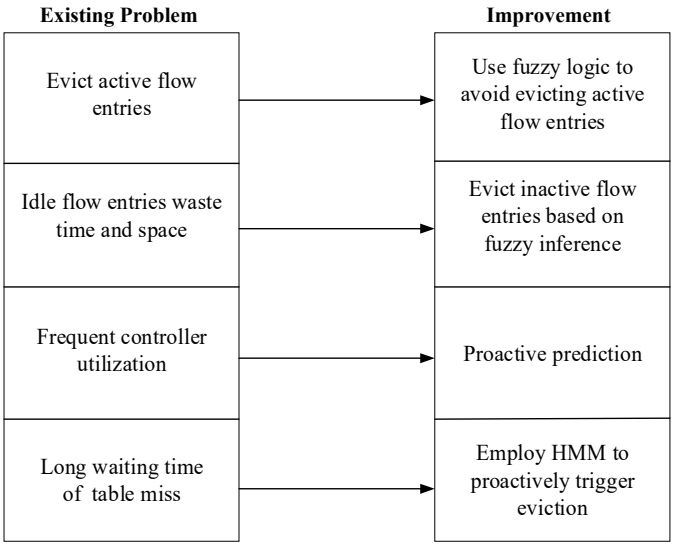


Fig. 3. The improvements made by the proposed scheme.

- The miss probability of a flow entry is predicted using the HMM. With four hidden states and three observable states for each entry, the miss probability can be accurately predicted.
- Fuzzy logic is employed to choose the victim entry based on the key characteristics of the entries modeled by a new notion of eviction index which is decided based on the match priority, idle time, and the number of unmatched flows.
- The proposed proactive eviction approach effectively accelerates the update of flow table and improves the match rate. It triggers the eviction of an entry before the occurrence of table miss, and thus increases the speed of flow management.

The rest of the paper is structured as follows. Section 2 discusses the work related to the management of SDN flow table, and Section 3 presents the proposed scheme of flow entry eviction based on HMM and fuzzy logic. In Section 4 the result of performance evaluation is given, and finally, Section 5 concludes the paper with some remarks.

2. RELATED WORK

2.1 Terminology

The terminologies used in the flow table management of SDN are defined as follows:

Definition 1 (Flow Entry)**:** This is the basic component of a flow table, where the number of flow entries depends on the size of the switch. Each flow entry consists of the field for match (Match Field), guideline for matched packet (Instructions), the number of matched packets (Counter), maximum amount of time or idle time (Timeout) before the entry is evicted, etc. Table 1 lists the basic fields of a flow entry.

Table 1. The basic fields of a flow entry.

Match Field	Priority	Counter	Instruction	Timeout	Cookie

Definition 2 (Stale Entry)**:** A flow entry may still exist in the flow table even after its usage is over, which is called stale entry. The holding time of stale entry is wasted time.

Definition 3 (Idle Timeout, T_I)**:** This is a time interval of a flow entry from the time of its packet arrival to the removal. If the idle timeout is up, the flow entry is evicted. The time to live for a flow entry is denoted by T_H.

Definition 4 (Match priority, *MP*)**:** *MP* is the priority of a flow entry to be matched. If a flow matches several flow entries, one of the largest *MP* is chosen.

Definition 5 (Idle time, *IT*)**:** *IT* of an entry denotes the period from the last match to the present time. The longer *IT*, the less the flow entry is useful.

Definition 6 (The number of unmatched flows, *NU*)**:** *NU* is the number of unmatched flows with the given flow table. The length of the unmatched time of a flow entry is from its entrance to the table to the present time.

Definition 7 (Eviction index, *EI*)**:** It represents the likeliness of a flow entry not to be used in the future. The larger *EI*, the more probably the flow entry is evicted.

2.2 Management of Flow Table

The limited size of flow table and declining performance with the increase of stale entry inspired the researchers to conceive effective methods of flow eviction. In OpenFlow network, a timeout mechanism is employed to flush the flow entry. In recent works, different values of timeout are used, ranging from 5 seconds [13] to 60 seconds as in DevoFlow [14]. If the timeout value is fixed, however, dynamically changing flows cannot be effectively handled. In order to resolve this issue, Adaptive Hard Timeout Method (AHTM) was proposed [15] where the *M*/*G*/*C*/*C*/FCFS queuing system is used to model the flow table. Using the model, the truncation time and blocking probability were analyzed to decide the optimal value of the hard timeout.

In [18] another scheme was proposed targeting dynamic hard timeout. It considers the flow duration time, flow type, and utilization of flow table to decide the value of hard timeout. Here the flow is classified into four kinds; short-small, long-small, short-large, and long-large flow. SmartTime [19] combined the adaptive idle timeout scheme with a proactive eviction strategy and utilized the Ternary Content Addressable Memory (TCAM) to minimize the number of table misses. [20] also proposed a scheme dynamically adjusting the idle timeout, where the controller collects the traffic information of the switches. According to the information, the idle timeout of each flow is predicted, and the timeout value is dynamically adjusted without modifying the switch configuration.

In [21] a flow entry eviction scheme using a Least-Recently-Used (LRU) caching algorithm was proposed. When a flow entry becomes idle, the counter value is reset to zero rather than eviction. When its counter value exceeds a threshold, the flow entry is evicted. Through this method, the switch can limit the number of idle flow entries. The authors of [22] proposed a flow table replacement algorithm. It estimates the priority of the flow entry using the information on the usage of switch link, number of matched flow entries, and idle timeout. The lower priority, the more likely the flow entry is evicted.

In DIFANE [23] a distributed flow management architecture was proposed where the rules are assigned to the authority switches, and all data traffic are handled via a fast path. DIFANE efficiently handles wildcard rules and quickly reacts to network dynamics such as policy and topology change and host mobility. Instead of a timeout value assigned to each flow entry, a space-efficient data structure, Multiple Bloom Filters (MBF), is used to log the data in [24]. The importance of each flow entry is encoded in MBF, which increases upon a match. When a table miss occurs, the entry of the lowest importance value is evicted. [25] proposed a dynamic flow scheduling method including flow-set routing and rerouting to perform load balancing and reduce the number of installed flow entries. To achieve balanced flow table usage, JumpFlow [8] was proposed to employ VLAN identifier (VID) in the packet header to deliver the routing information. In [26] an online routing scheme was utilized to maximize the network performance. In particular, it evicts idle flow entries according to the condition of real-time usage of each switch. [27] proposed a dynamic routing scheme to solve the problem of flow table overflow and inefficient bandwidth allocation by classifying flow and adaptively selecting routing paths for them. Flowmaster [16] uses learning predictor based on the Markov model to identify the flow entries expected to become idle. When a new flow arrives, an existing flow entry is evicted based on the number of transitions and transition probability [28], where the timeout mechanism is also used. In [17], a proactive approach is proposed based on the prediction of the pro-

bability of matching of the entries using HMM.

Various attempts have been made to maximize the efficiency of the eviction operation with the flow table of SDN while considering the scalability. One of the methods is to use fuzzy logic, which utilizes the methodology of computing with linguistic words in place of numbers [29]. Human employ words in the process of computing and reasoning, without an explicit use of any measurement. Fuzzy logic accurately enables intuitive reasoning, and thus reduces the gap between human thinking and mathematical reasoning [30]. Due to such merits, it has been used in the solution of diverse problems including flow eviction [16] and intrusion prediction and prevention [31]. We next present the proposed scheme, which further reduces the number of table misses using the HMM and fuzzy logic.

3. THE PROPOSED SCHEME

In this section the proposed scheme is presented, and its flowchart is illustrated in Fig. 4. As shown in the figure, the miss probability of a flow entry is predicted using the forward algorithm of HMM. If the probability exceeds the preset threshold, the eviction mechanism is triggered without waiting until the occurrence of an actual table miss. For eviction, fuzzy logic is used to choose the victim entry based on the key characteristics of the entries modeled by a new notion of eviction index. The entry of the largest eviction index is evicted to minimize the number of table misses.

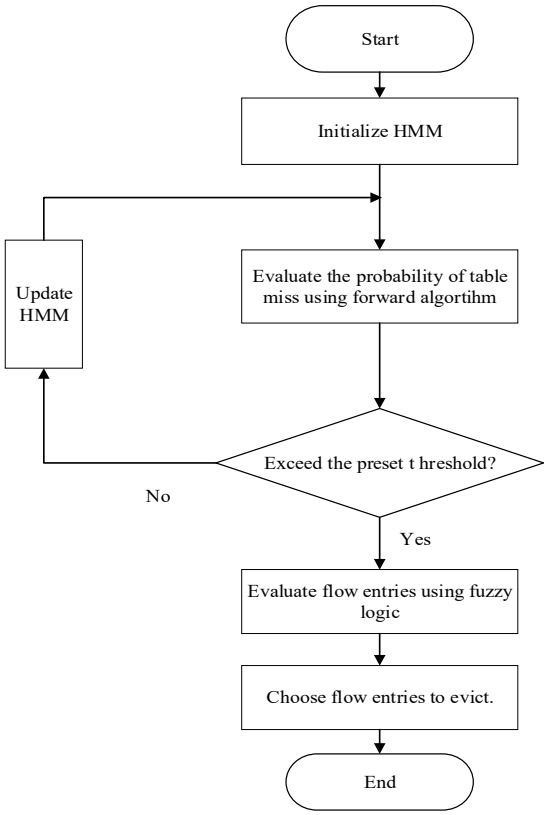


Fig. 4. The flowchart of the proposed scheme.

3.1 Prediction of Matching Probability

3.1.1 HMM

A hidden Markov model is a finite-state discrete-time first-order Markov model of two components, a homogeneous Markov chain with hidden states and a group of emission probabilities with the observable states. An HMM, represented as $\theta = (A, B, \pi)$, consists of the following elements:

- N hidden states of the Markov model denoted as $Q = \{q_0, q_1, q_2, ..., q_{(N-1)}\}$. X_t is the hidden state at time t.
- A set of M observable states denoted as $V = \{v_0, v_1, \ldots, v_{(M-1)}\}$. O_t denotes the observable state at time t.
- The state transition probability matrix, $A = \{a_{ij}\}$, is an $N \times N$ matrix with

$$a_{ij} = P(q_j \text{ at time } -(t+1) \mid q_i \text{ at time } -t),$$

$$a_{ij} \geq 0, \sum_{j=1}^{N} a_{ij} = 1, i, j = 0, ..., (N-1). \tag{1}$$

- The emission probability matrix, $B = \{b_j(k)\}$, is an $N \times M$ matrix with

$$b_j(k) = P(v_k \text{ at time } -t \mid q_i \text{ at time } -t). \tag{2}$$

- The initial hidden state distribution, $\pi = \{\pi_i\}$ $(i = 0, \ldots, (N-1))$, where $\pi_i = P(X_0 = q_i)$.

A generic HMM is illustrated in Fig. 5, where $X_i (i = 0, 1, \ldots)$ is the sequence of hidden states. The Markov process, located above the observable states, is represented by the current state and matrix A. Only O_i can be observed, which is represented by the hidden states of the Markov process and matrix B.

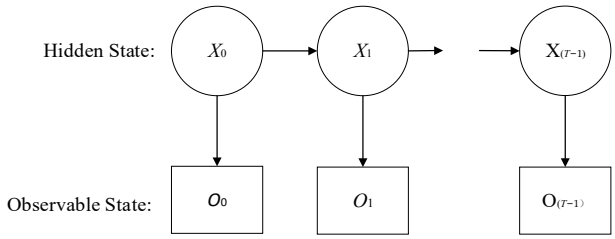


Fig. 5. The structure of hidden Markov model.

There exist three ways of solving the problems using HMM as follows.

- Given the model $\theta = (A, B, \pi)$ with a sequence of observation O, $P(O|\theta)$ is found. The forward algorithm is used to solve this problem.
- Given $\theta = (A, B, \pi)$ and O, an optimal state sequence for the underlying Markov process is found. In other words, the hidden part of the HMM is uncovered. The backward algorithm is used to solve this problem.

- Given O and the dimension N and M, the model $\theta = (A, B, \pi)$ maximizing the probability of O is found. This can be regarded as the training of a model to best suit the observed data. Alternatively, this can also be viewed as a discrete hill climb problem on the parameter space denoted by A, B, and π. Baum-Welch algorithm is often used as a solution.

3.1.2 The proposed HMM

Let $X = \{X_0, X_1, \ldots, X_{(n-1)}\}$ be a set of possible states of the system, while the hidden states are checked every Δt. Here X_i represents the state of the flow table at the time segment of $i\Delta t$, and $n\Delta t$ is the interval of updating the HMM. There could be four states for the hidden state, including *Normal* (*NO*) of no matching activity, *Match Check* (*MC*) of searching the flow table to match the flow entry, *Match in-Process* (*MP*) of processing the matched flow entry in the switch, and *Failure of Match* (*FM*) of no matched flow entry in the flow table. To simplify the notation, 0, 1, 2, and 3 are used to denote *NO*, *MC*, *MP*, and *FM*, respectively. X_0 is the initial state decided based on the current state.

Let $O = \{O_0, O_1, \ldots, O_{(N-1)}\}$ be a set of observable states. O_i indicates the observable state at the time $i\Delta t$. Three observable states are possible including *No matching activity* (*NM*), *Matching activity* (*MA*), and *Sending the packet-in* (*PI*) after table-miss. Again, 0, 1, and 2 are used to denote *NM*, *MA*, and *PI*, respectively. O_0 is the initial state. Figs. 6 and 7 show the structure and the state transition of the HMM of the proposed scheme, respectively.

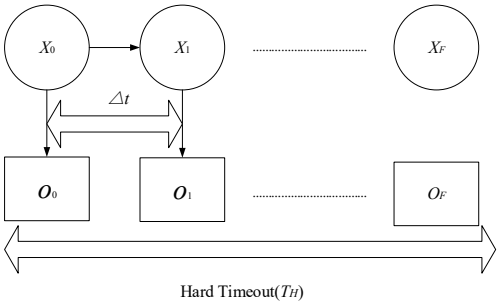


Fig. 6. The structure of the proposed HMM.

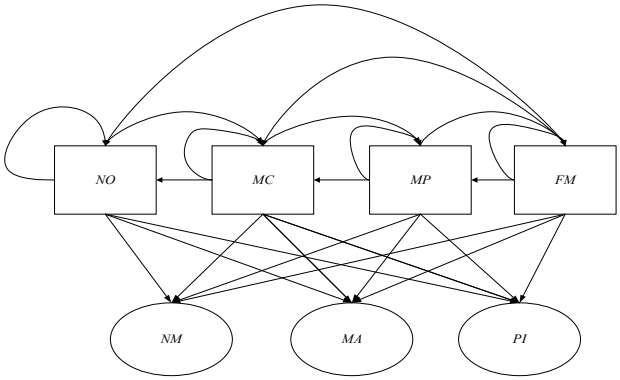


Fig. 7. The state transition diagram of the proposed HMM.

3.1.3 The operation

For the estimation of matching probability of an entry based on the HMM, the probability of the observable state sequence is used. Here three steps are involved; initialization, forwarding procedure, and final decision.

Step 1: Initialization

In order to calculate the probability required by the forward algorithm, $\theta = (A, B, \pi)$ needs to be obtained. Matrix A and B are decided by simulating the SDN handling a large number of packets. Notice that A is a 4×4 matrix, and B is a 4×3 matrix.

The hidden states and observable states are updated after every n seconds. Since the number of hidden states is 4, the probability of the states is initialized as [0.25 0.25 0.25 0.25]. The statistics of the packets are collected to increase the accuracy of the model, and the update interval is modified. The initial state is *MP*.

Step 2: Forwarding

First, the following is defined.

$$\alpha_t(i) = P(O_0, O_1, \ldots, O_t, X_t = q_i)\ (t = 0, 1, \ldots, (n-1)) \tag{3}$$

where $\alpha_t(i)$ is the probability of the partial observation sequence up to time $t\Delta t$, with the ith hidden state at time $t\Delta t$. n is the interval of update of the HMM model. In the beginning, the initial probability is obtained as follows.

$$\alpha_0(i) = \pi_i b_i(O_0) \tag{4}$$

where π_i is the initial probability of the ith hidden state.

Let $X = \{X_0, X_1, X_2, \ldots, X_{(n\text{-}1)}\}$ be the hidden state sequence. Then

$$P(O \mid X, \theta) = b_{x_0}(O_0) b_{x_1}(O_1) \ldots b_{x_{(n-1)}}(O_T) \tag{5}$$

and using π and A the probability of the hidden state sequence is

$$P(X \mid X, \theta) = \pi_{x_0} a_{x_0 x_1} a_{x_1 x_2} \ldots a_{x_{T-1} x_T}. \tag{6}$$

The joint probability of X and O is denoted as follows,

$$P(O, X \mid \theta) = \frac{P(O \cap X \cap \theta)}{P(\theta)}, \tag{7}$$

$$P(O, X \mid \theta) P(X \mid \theta) = \frac{P(O \cap X \cap \theta)}{P(\theta)} \cdot \frac{P(X \cap \theta)}{P(\theta)} = \frac{P(O \cap X \cap \theta)}{P(\theta)}. \tag{8}$$

Therefore,

$$P(O, X \mid \theta) = P(O \mid X, \theta)\, P(X \mid \theta). \tag{9}$$

By summing up the probability of all possible hidden state sequence, the following is obtained.

$$P(O|\theta) = \sum_X P(O, X|\theta) = \sum_X P(O|X,\theta)P(X|\theta)$$
$$= \sum_X \pi_{x_0} b_{x_0}(O_0) a_{x_0 x_1} b_{x_1}(O_1) a_{x_1 x_2} \ldots a_{x_{T-1} x_T} b_{x_T}(O_T) \quad (10)$$

For each time step t, $t = 1,2,\ldots, (n-1)$, the partial probability $\alpha_t(j)$ is calculated for each state.

$$\alpha_t(j) = b_j(O_t) \sum_{i=0}^{3} \alpha_{t-1}(i) a_{ij} \quad (11)$$

From Eq. (11) it is clear that $P(O|\theta)$ is the sum of the probabilities of $\alpha_t(i)(i = 0, \ldots, (N-1))$.

$$P(O|\theta) = \sum_{j=0}^{3} \alpha_t(j) \quad (12)$$

After the forward algorithm is applied, the probability of the observed state sequence is obtained. Here each observed state is *PI* in the given observed state sequence.

Procedure: Calculation of $P(O|\lambda)$.

Input:
(1) Observation sequence $O = \{O_0, \ldots, O_T\}$;
(2) HMM $\lambda = (A, B, \pi)$.
Output: the probability $P(O|\lambda)$ generated by observation sequence O of HMM.
Begin:
(1) Calculate the probability of O_0:
$\alpha_0(i) = \pi_i b_i(O_0)$.
(2) Calculate the probability of observation sequence O and $X_{t+1}(=q_j)$.
(a) at time t, calculate the probability of alert sequence O and $X_t(=q_i)$: $a_t(i)$.
(b) at time $t + 1$, calculate the probability of intent sequence O generated by HMM: λ, $X_t(= q_i)$, and $X_{t+1}(= q_j)$:

$\alpha_{t+1}(j) = [\sum_{i=1}^{N} \alpha_t(i) a_{ij}] b_j(O_{t+1})$ where $0 \le t \le (T-1)$; $0 \le j \le (N-1)$.

(3) Calculate the probability of the intended sequence O generated by HMM.

$$P(O|\lambda) = \sum_{j=0}^{N-1} \alpha_t(j)$$

(4) Return $P(O|\lambda)$.
End;

Step 3: Final Decision

Let p_{TH} be the preset threshold for table miss. The value obtained in Step 2 is compared with p_{TH}. If it is larger than p_{TH}, table miss will highly likely occur in the next $n\Delta t$,

and thus eviction might be needed. Notice that the proposed scheme still incorporates the basic hard timeout mechanism. Once the hard timeout for a flow entry is up without matching, it is evicted from the flow table.

3.2 Flow Eviction with Fuzzy Logic

3.2.1 Fuzzy logic

Fuzzy logic is a tool providing an efficient way for working with imprecise linguistic data. It is especially helpful for which there is no effective conventional model, or the model is too complicated. The proposed flow entry eviction scheme utilizes fuzzy logic to enhance the accuracy of prediction further.

Fuzzy logic is an improved Boolean logic based on the principle that there is a small portion that is certainly out of all things in the system [32]. Things are mainly uncertain, which can be treated through two attributes: random and fuzzy. The classical set describes a group consisting of some members which are identifiable. However, there exist many groups of members which cannot be explicitly recognized. The group featuring such characteristics is called a fuzzy set. The fuzzy set A, is a generalization of a crisp or Boolean set, which is denoted in a universe of discourse X. A has linguistic values, which define the fuzzy set using words. Such a word describes how a human expert perceives the linguistic variable X in the relationship with A. A is characterized by a membership function, $\mu_A(x)$, which evaluates the degree of similarity of an element x from X to A [33]. Its value is in [0,1], that is:

$$A = \{(x, \mu_A(x))|x \in X\},\ \mu_A(x): X \rightarrow [0, 1]. \tag{13}$$

Fuzzy logic can be used to decide or control a system based on the principle of fuzzy inference. It has two vital methods: Mamdani fuzzy inference and Sugeno fuzzy inference [30]. In this paper, the former is adopted, which is commonly used due to simple "min-max" operation. It consists of four stages as follows.

i) Fuzzification: in this stage, the experts consider the details concerning the input, output, and results. Linguistic variables are applied to explain each input and output variable. The fuzzy set and its membership function are determined by the linguistic variables. Then, fuzzy rules are formed to illustrate the relation between input and output.

ii) Fuzzy rule evaluation: the value of the membership function of each rule is computed using the following equation. The input and output are modeled with linguistic values, A_{ij} ($i = 1, 2, 3, \ldots, L$ and $j = 1, 2, 3, \ldots, n$) and B_i ($i = 1, 2, 3, \ldots, L$), respectively, where L is the number of fuzzy rules, and n is the number of discrete points of the fuzzy set.

$$\mu_i(x) = \min[\mu_{A_{i1}}(x_1), \mu_{A_{i2}}(x_2), \ldots., \mu_{A_{in}}(x_n), \mu_{B_i}(y)] \tag{14}$$

If the value of the membership function of any rule is zero, it is not considered in the decision. Otherwise, the value is used to truncate or measure the shape of the output membership function of the rule.

iii) Aggregation: the fuzzy set of the outputs obtained in stage ii) is combined by a union

operation of the rules. The max operator is used for the compensation between input variables as follows, where $\mu_B(y)$ is the aggregated output.

$$\mu_B(y) = \max\{\mu_{B_1}(y), \mu_{B_2}(y), \ldots, \mu_{B_L}(y)\} \tag{15}$$

iv) Defuzzification: the output fuzzy numbers are changed to crisp numbers based on the center of gravity (COG) approach of Eq. (16).

$$COG = \frac{\int_Y y \times \mu_B(y) dy}{\int_Y \mu_B(y) dy} \tag{16}$$

3.2.2 Fuzzification for eviction

The three indicators of flow entry eviction are match priority (δ), idle time, and the number of unmatched flows. The following fuzzy terms define them: very high, high, moderate, low, and very low, numbered from 1 to 5, respectively. Let us denote δ_h the highest match priority in the flow table. In order to select a flow entry for eviction, the eviction index (EI) of each entry is obtained as follows.

$$EI = (\delta_h - \delta) \times \text{Idle time} \times \text{number of unmatched flows} \tag{17}$$

The first term of EI indicates the relative priority in the table, with smaller value higher priority. It is evident that the larger the three indicators, the more probably the flow entry is evicted. The linguistic value of EI is high, moderate, and low.

3.2.3 Fuzzy inference

The Fuzzy Inference System (FIS) used to decide the eviction of a flow entry based on the reasoning process (rules) is shown in Fig. 8.

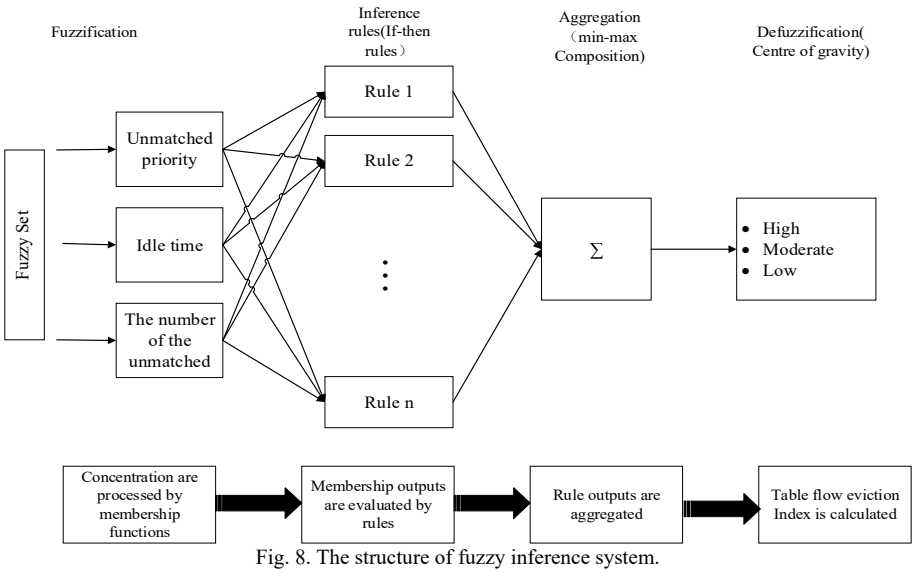


Fig. 8. The structure of fuzzy inference system.

The FIS is used to decide the eviction index using the fuzzy theory and thereby increases the effectiveness of the proposed scheme. Uncertainty is a distinct feature of FIS, which allows effective treatment of the parameters affecting the target operation, here eviction. Since the classification of the entries is ambiguous, it is needed to quantify the relative importance of each entry. Subjectivity refers to a specific interpretation of the aspect of experiences. The proposed *EI* reflects the subjectivity to properly handle various situations in flow entry eviction, influenced by different factors. The process is fulfilled using rules, which are included within the FIS. Both uncertainty and subjectivity are influential notions for effective assessment of *EI* in the proposed scheme.

Fuzzy inference is to formulate a mapping from a given input to an output using fuzzy logic [33]. The mapping provides a basis from which the decisions are made, or patterns are discerned. The process of fuzzy inference is handled in three phases: membership function, aggregation, and defuzzification.

The membership function is used to determine the membership level for the input. It transforms real data into a value in [0,1]; the most common functions are triangular, rectangular, trapezoidal, or Gaussian. In the proposed FIS, two types of membership functions are used for each flow entry: input membership function for match priority, idle time, and the number of unmatched flows, and output membership function for the *EI*. Gaussian membership function is utilized as the membership function as it statistically evaluates the elements affecting the eviction in [0,1] range. It is important to remark that *EI* of each flow entry is associated with one membership function. On the other hand, the Gaussian membership function defines the output transformation of the *EI* in three levels.

$$\mu(x) = e^{-\frac{(x-m)^2}{2k^2}} \tag{18}$$

Here x is an input, m is the central value, and k is the standard deviation.

To control the system, the scholars create the causal relations between the input and output; IF input THEN output. This is called a fuzzy rule. The number of fuzzy rules depends on the number of linguistic variables employed to define the input and output. Suppose that a system has n inputs and one output. The causal relation between them is shown with L rules. Let $x = [x_1, x_2, x_3, \ldots, x_n]$ be a set of values of the input and y an output value. A general form of ith rule of the fuzzy rule is as follows.

$$\text{If } (x_1 = A_{i1}) \text{ and } (x_2 = A_{i2}) \text{ and } \ldots \text{ and } (x_n = A_{in}), \text{ then } (y = B_i) \tag{19}$$

For flow entry eviction, the inference rules are built considering the consequence. For accurate decision, it is essential to synthesize the indicators affecting the eviction. The synthesis is made based on the corresponding fuzzy composition. Based on mathematical analysis, it has been proved that the min-max composition is accurate concerning the grouping of the results around the final equivalence for the establishment of the relationships.

Once the ruleset has been processed, an integration of the results, μ_R, needs be done.

$$\mu_R = \min\{\mu^i_{MP}, \mu^i_{It}, \mu^i_{NU}, \mu^i_{EI}\} \tag{20}$$

where i, j, k, and l are the evaluated levels respectively; μ_{MP}, μ_{IT}, μ_{NU}, μ_{EI}, μ_R are the membership of match priority, idle time, number of unmatched flows, eviction index, and the

integrated results, respectively. This expression denotes the rule antecedent. The output membership functions for *EI* are matched and truncated according to the rule as follows:

$$\mu_{R\_out} = \max\{\mu_{R_1}, \mu_{R_1}, ..., \mu_{R_n}\} \tag{21}$$

where n is the number of the fuzzy rules.

The final step of the FIS for flow entry eviction is the defuzzification process, where the *EI* is computed using the most commonly adopted centroid method. It obtains the center of the area under the curve formed by the fuzzy output function using Eq. (22).

$$\overline{EI} = \frac{\int x \cdot \mu_{R\_out} dx}{\int \mu_{R\_out} dx} \tag{22}$$

Then the diagram of the membership function of the *EI* is checked. If $\overline{EI}$ is higher than "high", the flow entry corresponding to this value is evicted from the flow table. Next, the proposed scheme is evaluated.

4. PERFORMANCE EVALUATION

In this section computer simulation is conducted to evaluate the effectiveness of the proposed flow entry eviction scheme in terms of prediction accuracy, match rate, and the number of table misses.

4.1 Simulation Setting

The simulation is conducted on Intel Core i7 process, 2.50GHz PC with 16GB RAM and Matlab R2015a. The results of the proposed scheme are also compared with three representative eviction techniques; the HMM-based scheme (HS) [17], the Flowmaster scheme [16] employing dynamically adjusted timeout, and the standard Hard timeout (HT) algorithm.

The match rate of a scheme is obtained by dividing the number of successful matches by the number of flows entering the switch after the last eviction. The flow is generated according to an exponential distribution with $\lambda = 1$, and the contents of the flow entries are decided randomly. The size of each flow is randomly decided between 100KB and 20MB.

A virtual network has been deployed using Mininet with the following elements: Open vSwitch, opendaylight controller, and end-host nodes with the service provider (SP) enabled. The host nodes, switch, and controller were running in Ubuntu 16.04 LTS. A switch connected to two hosts was implemented as shown in Fig. 9. The switch is linked to a controller where the opendaylight is used to control and coordinate the whole networking operation.

4.2 Experiment Results

Let n_f and n_e denote the number of flows tested and flow entries in a flow table, respectively. p_{TH} is decided based on the training of the previous values. In Fig. 10 the match probability of the proposed scheme is compared with other schemes for different (n_f, n_e)

values. Here T_H = 50s. Observe from the figure that the match probability of the proposed scheme with (100, 50) is always higher than the other two schemes. This demonstrates that the proposed scheme based on the HMM and fuzzy logic is effective in managing the flow entry.

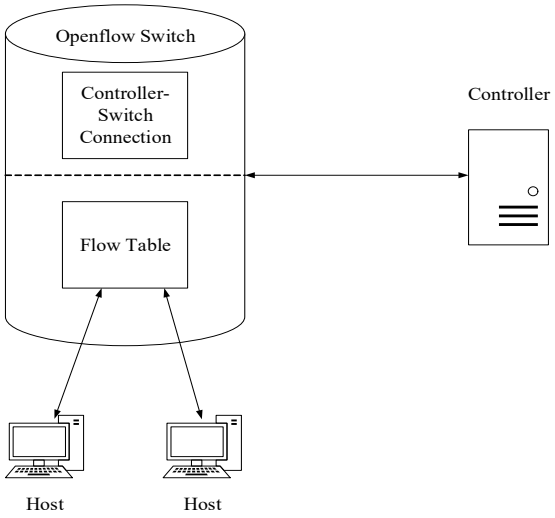


Fig. 9. The structure of simulation for SDN.

The data in the middle of the figure shows the match probability with a larger number of flows of 200 but the same number of entries of 50. The proposed scheme consistently displays a higher match probability. It is also observed that the match probability is slightly smaller than those in the left because more flows are supported with the flow table of the same size. When the number of flow entries is added to 100 as in the right-hand side of the figure, the match probability also increases. Note that the proposed scheme consistently allows significantly higher match probability regardless of the operational conditions.

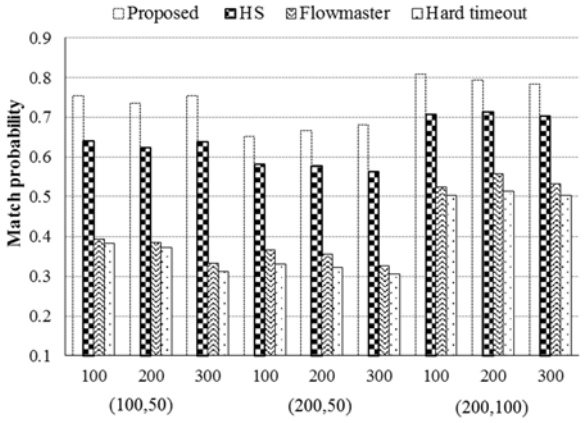


Fig. 10. The comparison of match probabilities with different (n_f, n_e) values.

Another performance metric is the number of table misses which is shown in Fig. 11. Note that the proposed scheme substantially decreases the occurrence of table misses compared with the other schemes through effective prediction of the access probability of the flow. Particularly, the number of table misses increases if n_f increases from 100 to 200 while n_e is unchanged as shown in the middle of the figure. However, the number of table misses of the proposed scheme is still much smaller than the other schemes. When the number of the flow entries is increased to 100, the number dramatically decreases. Notice that the proposed scheme consistently outperforms the other schemes regardless of the size of the flow table.

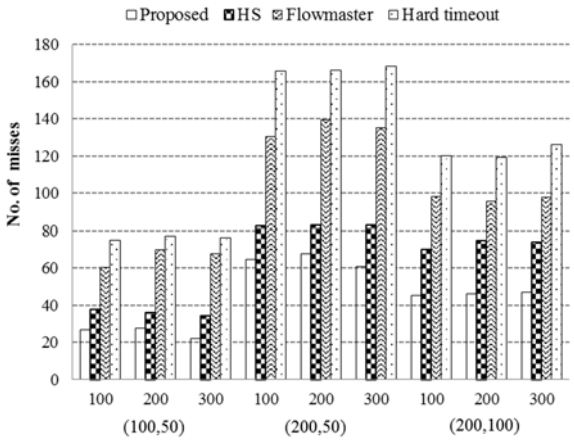
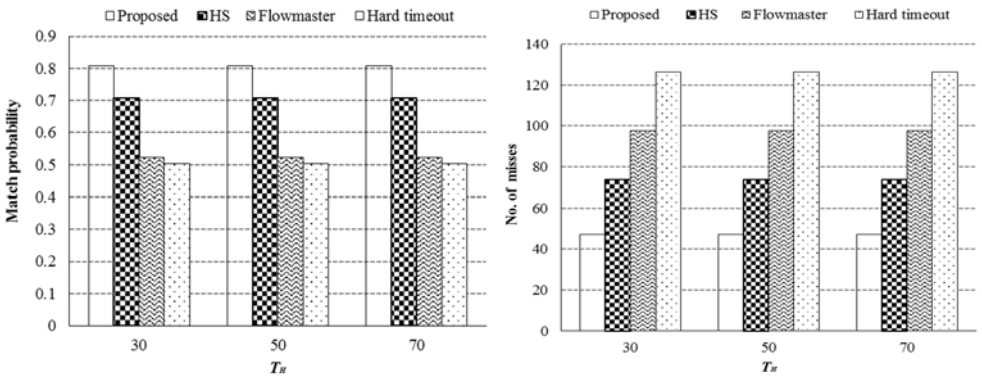


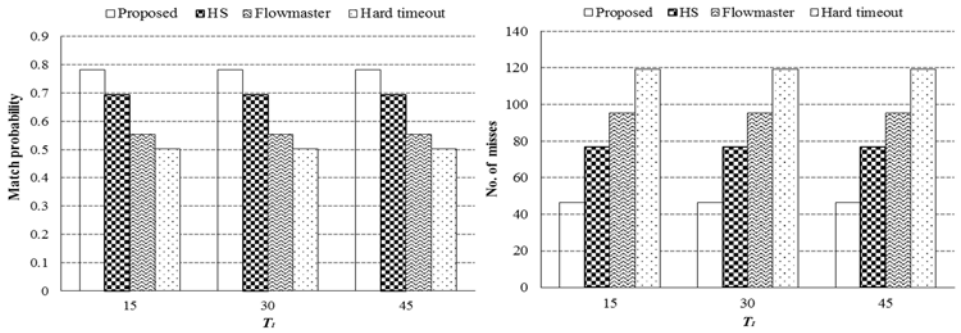
Fig. 11. The comparison of the number of table misses of the four schemes.



(a) Match probability. (b) Number of misses.

Fig. 12. The comparison of the schemes with varying T_H.

The effect of T_H and T_I on the matching probability and the number of table misses are studied in Figs. 12 and 13, respectively, with 300 matches and (200, 100). Observe from the figures that the proposed scheme substantially outperforms the others regardless of the T_H and T_I value.



(a) Match probability. (b) Number of misses.

Fig. 13. The comparison of the schemes with varying T_I.

Fig. 14 shows the accuracy of the proposed scheme in predicting table miss, where the prediction occurs upon table miss. Here $T_I = 10$. Note from the figure that the proposed scheme consistently allows accurate prediction in various operational conditions.

Next the effect of p_{TH} on the number of misses is investigated with 300 matches and (200, 100). Observe from Fig. 15 that the number of misses increases with the increase of p_{TH}. This is because the larger the threshold, the fewer flow entries are predicted to be evicted causing more table misses.

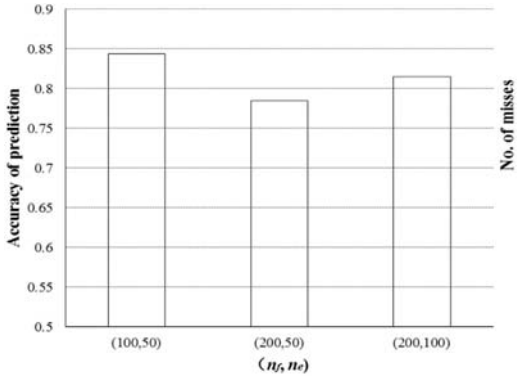


Fig. 14. The prediction accuracies of the proposed scheme.

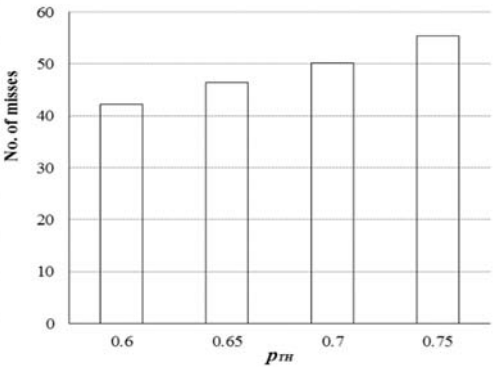


Fig. 15. The comparison of the number of table misses with varying p_{TH}.

5. CONCLUSIONS

In this paper a novel flow entry eviction scheme for SDN has been proposed. It employs HMM to evaluate the probability of the occurrence of table misses. If the probability exceeds the preset threshold, fuzzy logic is used to select the victim entry before the actual table miss occurs so that the occurrence of table misses can be minimized. This accelerates the update of flow table and improves the probability of matching of flow entries by utilizing the statistical information of the table match. The proposed scheme was simulated using Opendaylight and Matlab, and it turns out to substantially enhance the efficiency of flow table management in terms of match probability, the number of table misses, the con-

troller overhead, and CPU consumption rate compared to the existing schemes.

In the future the prediction accuracy of the proposed scheme will be further enhanced by investigating the factors influencing the performance of prediction including idle timeout and threshold, p_{TH}. In addition, the HMM model will be refined probably with different state definition. The analytical models estimating the target performance metrics will also be developed to properly decide the value of various operational parameters allowing maximum performance. In the proposed scheme fuzzy logic was employed to dynamically select the entries to be evicted, and in the future a machine learning technique such as reinforcement learning will be investigated for it. The usage of machine learning technique will also be considered for proactive flow rule installation.

ACKNOWLEDGEMENTS

This work was partly supported by Institute for Information and Communications Technology Promotion (IITP) grant funded by the Korea government (MSIT) (No. 2016-0-00133, Research on Edge computing via collective intelligence of hyperconnection IoT nodes), Korea, under the National Program for Excellence in SW supervised by the IITP (Institute for Information and Communications Technology Promotion) (2015-0-00914), Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education, Science and Technology (2016R1A6A3A11 931385, Research of key technologies based on software defined wireless sensor network for realtime public safety service, 2017R1A2B2009095, Research on SDN-based WSN Supporting Real-time Stream Data Processing and Multiconnectivity), the second Brain Korea 21 PLUS project.

REFERENCES

1. B. Nunes, *et al.*, "A survey of software-defined networking: Past, present, and future of programmable networks," *IEEE Communications Surveys & Tutorials*, Vol. 16, 2014, pp. 1617-1634.
2. W. Xia, *et al.*, "A survey on software-defined networking," *IEEE Communications Surveys & Tutorials*, Vol. 17, 2015, pp. 27-51.
3. I. F. Akyildiz, *et al.*, "A roadmap for traffic engineering in SDN-OpenFlow networks," *Computer Networks*, Vol. 71, 2014, pp. 1-30.
4. U. Javed, *et al.*, "A stochastic model for transit latency in OpenFlow SDNs," *Computer Networks*, Vol. 113, 2017, pp. 218-229.
5. J. Mao, *et al.*, "Efficient mismatched packet buffer management with packet order-preserving for OpenFlow networks," *Computer Networks*, Vol. 110, 2016, pp. 91-103.
6. A. Lara, *et al.*, "Network innovation using openflow: A survey," *IEEE Communications Surveys & Tutorials*, Vol. 16, 2014, pp. 493-512.
7. P. T. Congdon, *et al.*, "Simultaneously reducing latency and power consumption in openflow switches," *IEEE/ACM Transactions on Networking*, Vol. 22, 2014, pp. 1007-1020.
8. Z. Guo, *et al.*, "JumpFlow: Reducing flow table usage in software-defined networks," *Computer Networks*, Vol. 92, 2015, pp. 300-315.
9. H. Kim and N. Feamster, "Improving network management with software defined net-

working," *IEEE Communications Magazine*, Vol. 51, 2013, pp. 114-119.
10. G. Xu, *et al.*, "Bandwidth-aware energy efficient flow scheduling with SDN in data center networks," *Future Generation Computer Systems*, Vol. 68, 2017, pp. 163-174.
11. C. Y. Hsu, *et al.*, "A flow-based method to measure traffic statistics in software defined network," in *Proceedings of the Asia-Pacific Advanced Network*, Vol. 38, 2014, pp. 19-22.
12. M. Karakus and A. Durresi, "Quality of service (QoS) in software defined networking (SDN): A survey," *Journal of Network and Computer Applications*, Vol. 80, 2017, pp. 200-218.
13. N. Gude, *et al*., "NOX: towards an operating system for networks," *ACM SIGCOMM Computer Communication Review*, Vol. 38, 2008, pp. 105-110.
14. A. R. Curtis, *et al.*, "DevoFlow: Scaling flow management for high-performance networks," in *ACM SIGCOMM Computer Communication Review*, 2011, pp. 254-265.
15. L. Zhang, *et al.*, "AHTM: Achieving efficient flow table utilization in software defined networks," in *Proceedings of IEEE Global Communications Conference*, 2014, pp. 1897-1902.
16. K. Kannan, *et al.*, "Flowmaster: Early eviction of dead flow on sdn switches," in *Proceedings of International Conference on Distributed Computing and Networking*, 2014, pp. 484-498.
17. G. Huang and H. Y. Youn, "Proactive eviction of flow entry for SDN based on hidden Markov model," *Frontiers of Computer Science*, No. s11704-018-8048-2, 2018.
18. L. Zhang, *et al.*, "TimeoutX: An adaptive flow table management method in software defined networks," in *Proceedings of IEEE Global Communications Conference*, 2015, pp. 1-6.
19. A. Vishnoi, *et al.*, "Effective switch memory management in OpenFlow networks," in *Proceedings of the 8th ACM International Conference on Distributed Event-Based Systems*, 2014, pp. 177-188.
20. T. Kim, *et al.*, "A dynamic timeout control algorithm in software defined networks," *International Journal of Future Computer and Communication*, Vol. 3, 2014, pp. 331-336.
21. E. D. Kim, *et al*., "Flow table management scheme applying an LRU caching algorithm," in *Proceedings of International Conference on Information and Communication Technology Convergence*, 2014, pp. 335-340.
22. D. Kim, *et al*., "An efficient flow table replacement algorithm for SDNs with heterogeneous switches," in *Proceedings of the 7th International Conference on Information Technology and Electrical Engineering*, 2015, pp. 300-303.
23. M. Yu, *et al.*, "Scalable flow-based networking with DIFANE," *ACM SIGCOMM Computer Communication Review*, Vol. 41, 2011, pp. 351-362.
24. R. Challa, *et al.*, "Intelligent eviction strategy for efficient flow table management in openflow switches," in *Proceedings of IEEE NetSoft Conference and Workshops*, 2016, pp. 312-318.
25. Z. Guo, *et al.*, "Dynamic flow scheduling for power-efficient data center networks," in *Proceedings of IEEE/ACM 24th International Symposium on Quality of Service*, 2016, pp. 1-10.
26. Z. Guo, *et al.*, "STAR: Preventing flow-table overflow in software-defined networks," *Computer Networks*, Vol. 125, 2017, pp. 15-25.
27. Z. Guo, *et al.*, "Balancing flow table occupancy and link utilization in software-

defined networks," *Future Generation Computer Systems*, Vol. 89, 2018, pp. 213-223.
28. K. Yoshioka, *et al.*, "Routing method with flow entry aggregation for software-defined networking," in *Proceedings of International Conference on Information Networking*, 2017, pp. 157-162.
29. L. Suganthi, *et al.*, "Applications of fuzzy logic in renewable energy systems – a review," *Renewable and Sustainable Energy Reviews*, Vol. 48, 2015, pp. 585-607.
30. F. Valdez, *et al.*, "A survey on nature-inspired optimization algorithms with fuzzy logic for dynamic parameter adaptation," *Expert Systems with Applications*, Vol. 41, 2014, pp. 6459-6466.
31. K. Haslum, *et al.*, "Dips: A framework for distributed intrusion prediction and prevention using hidden markov models and online fuzzy risk assessment," in *Proceedings of the 3rd International Symposium on Information Assurance and Security*, 2007, pp. 183-190.
32. P. Vonglao, "Application of fuzzy logic to improve the Likert scale to measure latent variables," *Kasetsart Journal of Social Sciences*, Vol. 38, 2017, pp. 337-344.
33. J. J. Carbajal-Hernández, *et al.*, "Assessment and prediction of air quality using fuzzy logic and autoregressive models," *Atmospheric Environment*, Vol. 60, 2012, pp. 37-50.

Gan Huang received the B.S. in Electronic and Information Engineering from Chuzhou University, China, in 2012, and the M.S. from in Computer Science from Anhui Polytechnic University, China, in 2016. Currently, he is a Ph.D. student at the Department of Electrical and Computer Engineering, Sungkyunkwan University. His research interests include software-defined networking (SDN), ubiquitous and distributed computing.

.

Hee Yong Youn received the B.S. and M.S. in Electrical Engineering from Seoul National University, Seoul, Korea, in 1977 and 1979, respectively, and the Ph.D. in Computer Engineering from the University of Massachusetts at Amherst, in 1988. He had been Associate Professor of Department of Computer Science and Engineering, The University of Texas at Arlington until 1999. He is a Professor of College of Information and Communication Engineering and Director of Ubiquitous Computing Technology Research Institute, Sungkyunkwan University, Suwon, Korea, and he is presently visiting SW R&D Center, Device Solutions, and Samsung Electronics. His research interests include cloud and ubiquitous computing, system software and middleware, and RFID/USN. He has published numerous papers and received Outstanding Paper Award from the 1988 IEEE International Conference on Distributed Computing Systems, 1992 Supercomputing, IEEE 2012 International Conference on Computer, Information and Telecommunication Systems, and CyberC 2014. Prof. Youn has been General Co-Chair of IEEE PRDC 2001, International Conference on Ubiquitous Computing Systems (UCS) in 2006 and 2009, UbiComp 2008, CyberC 2010, Program Chair of PDCS 2003 and UCS 2007, respectively.