PAPER

Separating Predictable and Unpredictable Flows via Dynamic Flow Mining for Effective Traffic Engineering*

Yousuke TAKAHASHI[†], Keisuke ISHIBASHI[††a)], Masayuki TSUJINO[††], Noriaki KAMIYAMA[†††], *Members*, Kohei SHIOMOTO[††††], *Fellow*, Tatsuya OTOSHI[†††††], Yuichi OHSITA[†††††], *Members*, *and* Masayuki MURATA[†††††], *Fellow*

SUMMARY To efficiently use network resources, internet service providers need to conduct traffic engineering that dynamically controls traffic routes to accommodate traffic change with limited network resources. The performance of traffic engineering (TE) depends on the accuracy of traffic prediction. However, the size of traffic change has been drastically increasing in recent years due to the growth in various types of network services, which has made traffic prediction difficult. Our approach to tackle this issue is to separate traffic into predictable and unpredictable parts and to apply different control policies. However, there are two challenges to achieving this: dynamically separating traffic according to predictability and dynamically controlling routes for each separated traffic part. In this paper, we propose a macroflow-based TE scheme that uses different routing policies in accordance with traffic predictability. We also propose a traffic-separation algorithm based on real-time traffic analysis and a framework for controlling separated traffic with software-defined networking technology, particularly OpenFlow. An evaluation of actual traffic measured in an Internet2 network shows that compared with current TE schemes the proposed scheme can reduce the maximum link load by 34% (at the most congested time) and the average link load by an average of 11%.

key words: *software defined networking, routing, traffic engineering, macroflow, flow mining*

1. Introduction

Software-defined networking (SDN), particularly OpenFlow, has attracted increasing attention because of its potential to enable effective traffic engineering (TE). OpenFlow [2], [3] can control flow-level traffic defined by any combination of L2-L4 packet header information such as 5-tuple headers (source and destination addresses, source and destination ports, transport layer protocol). Software-defined networking(SDN)/OpenFlow allows instant control of the routes of flow traffic because the controller centrally controls all switches in the network.

Although the advent of SDN/OpenFlow makes it easy to enable TE, there are still some issues. One of the most important issues with current TE schemes is that their performance significantly depends on the accuracy of traffic prediction [4]–[6]. However, the size of traffic change has been drastically increasing in recent years due to the growth in various types of network services and it makes traffic prediction difficult. Under such circumstances, it is difficult for current TE schemes to improve the utilization efficiency of network resources. Although there are several TE schemes that cope with unpredictable traffic by not depending on traffic prediction [7], [8], such schemes are mainly focused on managing unpredictable traffic for congestion avoidance; thus, utilization efficiency might be still low.

Such limitations of current TE schemes are caused by controlling all traffic with a single routing policy even though various types of traffic are mixed in the network due to diversification of network services. Our idea to overcome such limitations is to separate traffic according to its characteristics and to control each separated traffic with a control policy suitable for these characteristics.

In this paper, we propose a macroflow-based TE scheme that applies different routing policies for different traffic groups separated according to their predictability. Our proposed TE scheme uses *macroflows* as a granularity of routing control (i.e., a match field of a flow entry). A macroflow is a flow group consisting of a number of 5-tuple flows, whose type is classified as predictable or unpredictable and is expressed by any combination of 5-tuples that allows wildcards. By doing so, our macroflow-based TE scheme is expected to achieve more effective routing than current TE schemes applying a single routing policy.

To enable our macroflow-based TE scheme, we also propose a *hierarchical multiple flow table architecture* and *macroflow-generating method*. The proposed architecture is introduced to meet the multiple objectives for macroflows with a small size of table. The macroflow-generation method finds predictable/unpredictable macroflows within a small number of entries from an enormous number of 5-tuple combinations within a practical amount of time.

Through simulation evaluation using actual NetFlow data and topology information of Internet2 [9], we show that our proposed scheme can reduce the maximum link load in a network at the most congested time and the average link load

Manuscript received February 24, 2017.
Manuscript revised June 28, 2017.
Manuscript publicized August 7, 2017.

[†]The author is with NTT Communications Corporation, Tokyo, 108-8118 Japan.
[††]The authors are with NTT Network Technology Laboratories, NTT Corporation, Musashino-shi, 180-8585 Japan.
[†††]The author is with Faculty of Engineering, Fukuoka University, Fukuoka-shi, 814-0180 Japan.
[††††]The author is with Faculty of Knowledge Engineering, Tokyo City University, Tokyo, 158-8557 Japan.
[†††††]The authors are with Graduate School of Information Science and Technology, Osaka University, Suita-shi, 565-0871 Japan.

[*]A preliminary version of this work appeared in IEEE ICC2016 [1].

a) E-mail: ishibashi.keisuke@lab.ntt.co.jp

DOI: 10.1587/transcom.2017EBT0001

in a network on average compared with current TE schemes.

In summary, our key contributions are:

- We propose a macroflow-based TE scheme that efficiently synthesizes different routing policies in accordance with traffic predictability to optimize traffic routes.
- We propose a hierarchical multiple-flow-table architecture to place and connect different types of flow tables hierarchically. It efficiently enables routing control to achieve multiple objectives within a limited flow table space.
- We also propose a macroflow-generation method based on an iterative improvement method. We evaluated the performance of this method using real traces. It aggregated hundreds of thousands of 5-tuple flows into tens of macroflows within five minutes. The generated predictable macroflows have traffic characteristics in which the time variation of traffic load is small.
- We evaluated the performance of our proposed scheme using real traces. The results show that our TE scheme can reduce the maximum link load in a network at the most congested time by 34% and the average link load in a network on average by 11% compared with current TE schemes.

This paper is organized as follows. In Sect. 2, we review previous studies on flow aggregation and TE. In Sect. 3, we give an overview of our macroflow-based TE scheme. In Sect. 4, we explain our hierarchical multiple-flow-table architecture and macroflow-generation method. We also discuss the use of two route-optimization methods for enabling our scheme. In Sect. 5, we discuss our evaluation of our scheme. In Sect. 6, we discuss the issues for realizing our proposed TE and the possibilities about improvement of our proposed TE methods. Finally, we conclude the paper and briefly comment on future work in Sect. 7.

2. Related Work

2.1 Flow Aggregating Methods

Various methods have been proposed to aggregate traffic flow [10], [11]. The objective with these methods is to find a cluster of attack flows in real time. These methods limit a solution space so as to specialize in real-time attack detection. Hence, they can only generate macroflows defined by a limited combination of L2-L4 packet headers. Our target is to generate macroflows for TE. It is difficult with such methods to aggregate flows in accordance with traffic characteristics within a limited number of flow entries.

Kamiyama et al. [12] proposed a method of aggregating microflows into a flow group to minimize variations in traffic volume. They found that using the flow group generated with their method as a routing granularity improves route stability in their simulation. However, it is difficult to use their method in OpenFlow networks because millions of flow entries are needed. Our macroflow-generation method generates implementable flow groups, namely macroflows, by a combination of 5-tuple packet headers within a limited flow table space.

2.2 Traffic Engineering Schemes

Many TE studies target the efficient use of network bandwidth. Representative TE schemes have been developed using Interior Gateway Protocol (IGP) [13] and multi-protocol label switching (MPLS) [14]. These TE schemes have limitations in both engineering granularity and adaptability to traffic fluctuation (spikes). For example, traffic routes are determined based on their destination routers [13] and their origin-destination router pairs [14]. In addition traffic routes are determined based on their predicted traffic volume. In this case, if one single spike flow is mixed in the traffic of an origin-destination router pair, all the traffic might be inappropriately routed. Recently, SDN techniques have been applied to achieve fine-grained TE and applied to control inter-datacenter networks [15], [16]. Software-defined-networking-based TE achieves both bandwidth allocation based on application-service priority and efficient routing to improve network utilization. In such networks, routing granularity can be configured statically since administrators have a large amount of information about target traffic, e.g., endpoints of flows are particular servers or there are only a few applications such as data backup. We focus on SDN-based TE for a network in which there are many heterogeneous hosts and applications, such as ISP backbone networks. To implement SDN-based TE in such networks, it is necessary to adaptively configure routing granularity via dynamic data mining of observed flow data.

3. Proposed Macroflow-Based Traffic Engineering Scheme

We propose a scheme that controls traffic at the granularity level of macroflows. Macroflows are generated by grouping 5-tuple flows whose origin-destination (OD) pairs are the same. Generated macroflows are labeled as predictable or unpredictable. We assume that traffic whose volume has changed drastically over the past period is unpredictable. Our TE scheme then calculates optimal routes for those macroflows based on their labels, i.e., predictable or not. As noted in Sect. 2, OD traffic is too coarse to use traffic characteristics, predictable or not. On the other hand, 5-tuple flows are too fine because predictability emerges when those flows are aggregated. In addition, it is infeasible to control each 5-tuple flow with a limited number of flow tables. By adopting route optimization with granularity of macroflows, we can take advantage of their characteristics for route optimization with a limited size of flow tables.

Our scheme synthesizes two different types of TE schemes: *prediction-based routing* that uses predicted traffic demand [4] and *load-balanced routing* without using traffic prediction [7], [8]. For predictable traffic, prediction-based routing will outperform load-balanced routing in terms of

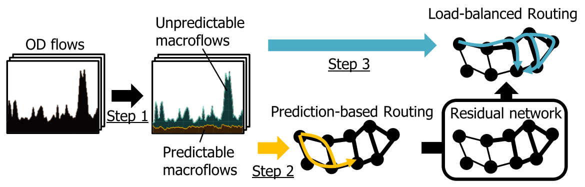


Fig. 1 Overview of proposed macroflow-based traffic engineering.

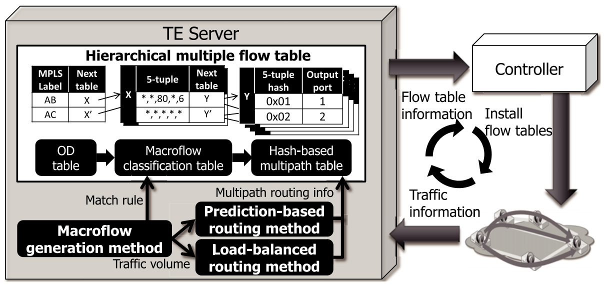


Fig. 2 Framework for enabling macroflow-based TE scheme.

routing efficiency.

For unpredictable traffic, however, load-balanced routing will outperform prediction-based routing for congestion avoidance. The key idea of our scheme is to use different TE schemes in accordance with the predictability of macroflows. Specifically, predictable macroflows are controlled by prediction-based routing, and unpredictable macroflows are controlled by load-balanced routing. An overview of our proposed scheme is illustrated in Fig. 1. Our proposed scheme is composed of the following three steps.

Step 1. Generating macroflows from 5-tuple microflows by grouping them into predictable and unpredictable ones.
Step 2. Computing optimal routing paths of predictable macroflows by using prediction-based routing method. Specifically, we predict the volume of predictable macroflows for the next time slot and solve optimization problems with the predicted volume of predictable macroflows as input.
Step 3. Computing routing paths of unpredictable macroflows by using the load-balanced routing method for the residual network after conducting Step 2.

4. Framework for Enabling Our Macroflow-based TE Scheme

To enable a macroflow-based TE, we adopt SDN/OpenFlow. Figure 2 shows our TE framework, which consists of SDN/OpenFlow switches, a controller, and TE server. The TE server, which is a key component of our architecture, periodically monitors NetFlow data (traffic information in Fig. 2). Then, it generates flow tables for each switch through three functions: hierarchical arrangement of multiple flow tables, macroflow generation, and route optimization. We explain the three functions in detail in the following subsection. Finally, the TE server sends the generated flow tables to the SDN/OpenFlow controller to update the flow tables of OpenFlow switches.

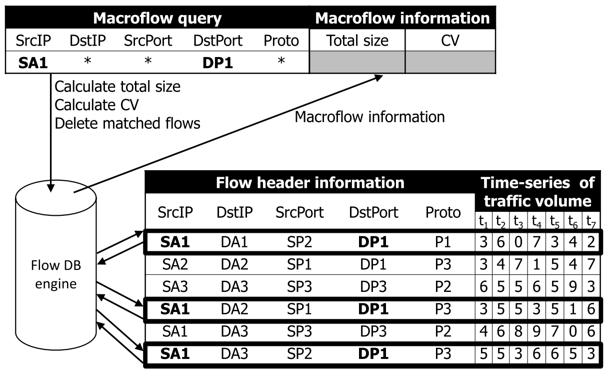
4.1 Hierarchical Arrangement of Multiple Flow Tables

To implement macroflow-based TE with SDN/OpenFlow, OD information, predictability information, and forwarding information of the arrived flow needs to be identified on the switch. These three types of information are described as follows.

- OD information
  It is necessary to ensure the controllability of traffic for TE. To use a common TE algorithm (e.g., multicommodity flow optimization [4], [17]) for route optimization, the OD information for traffic needs to be identified on a flow table.
- Predictability information
  To apply different routing policies in accordance with traffic predictability, it is necessary to classify traffic according to whether it is predictable with granularity of macroflows.
- Forwarding information
  It is important to ensure that traffic will be splittable at any size ratio. Traffic engineering algorithms that compute routing paths within a practical time solve the linear programming problem and output the optimal traffic splitting ratio among multiple paths. Hence, to install the calculated multipath routing information to the flow switches, traffic needs to be splittable into any ratio at the flow switches.

Since a macroflow initially generated with our scheme does not necessarily belong to a single OD pair, it must be split into each OD pair (requirement for OD information). In addition, to apply multi-path routing, it should be again split into any ratio calculated by the TE server (requirement for forwarding information). Meeting these requirements with a single flow table is inefficient. We therefore propose a *hierarchical multiple-flow-table architecture* to arrange three types of flow tables hierarchically; OD, macroflow, and multipath, as shown in Fig. 2. To construct our hierarchical multiple-flow-table architecture, we use the OpenFlow feature called "multiple tables" that enables an OpenFlow switch to repeatedly match flows to multiple flow rules on multiple flow tables. Our architecture provides flexible flow table management. Each flow table, which is separated by different objectives, can be updated individually.

We explain the proposed framework in Fig. 2. In the first flow table, flows are separated in accordance with their MPLS labels to identify an OD pair of a flow. The MPLS labels are attached to flows at the ingress router in advance. Next, flows are aggregated into macroflows in accordance with macroflow matching rules generated with our proposed macroflow generation method (explained below). Finally, flows are split into multiple output ports on the basis of the hash value of their flow header. The flow splitting ratio in this



Macroflow query					Macroflow information	
SrcIP	DstIP	SrcPort	DstPort	Proto	Total size	CV
SA1	*	*	**DP1**	*		

Flow header information					Time-series of traffic volume						
SrcIP	DstIP	SrcPort	DstPort	Proto	t_1	t_2	t_3	t_4	t_5	t_6	t_7
SA1	DA1	SP2	**DP1**	P1	3	6	0	7	3	4	2
SA2	DA2	SP1	DP1	P3	3	4	7	1	5	4	7
SA3	DA3	SP3	DP3	P2	6	5	5	6	5	9	3
SA1	DA2	SP1	**DP1**	P3	3	5	5	3	5	1	6
SA1	DA3	SP3	DP3	P2	4	6	8	9	7	0	6
SA1	DA3	SP2	**DP1**	P3	5	5	3	6	6	5	3

Fig. 3 Flow database for macroflow-generating method.

table is configured using the output of two route-optimization methods (explained below). Flows classified with the three types of flow tables meet all the requirements.

We briefly discuss the implementation of our proposed architecture. OpenFlow 1.1 (or above) supports MPLS labels in matching fields [3], and unequal flow splitting on the basis of the hash value of a 5-tuple, called "multipath function", is implemented in Open vSwitch [18]. OpenFlow 1.3 (or above) supports "group table", which also enables unequal splitting [3]. We confirmed that these functions on physical machines work as expected through a verification experiment.

4.2 Macroflow-Generating Method

Next, we describe our proposed macroflow-generation method. A macroflow is defined by any combination of 5-tuples that allows wildcards. Our goal is to generate predictable/unpredictable macroflows within a small number of entries from many 5-tuple flows. However, it is impractical with a mathematical optimization approach to find predictable/unpredictable macroflows from the enormous number of possible combinations of 5-tuples in terms of running time. We thus design a macroflow-generation algorithm for the proposed method that runs in an iterative improvement manner. In addition, to assist this algorithm, we implement a flow database illustrated in Fig. 3. This database stores 5-tuple flow information as header information and the time-series of its traffic volume. When the database receives a macroflow query including macroflow-header information and instructions, it finds flows matching the macroflow header then calculates and outputs the sum and coefficient of variance (CV) for the time-series of traffic volume of the matched flow.

The algorithm is shown in Algorithm 1. It consists of four steps: 1) summarizing flow data, 2) setting the initial candidate macroflows, 3) randomly selecting an operation from three types of basic operations, and 4) updating the candidate macroflows. After step 4 is completed, the algorithm checks whether the termination conditions have been achieved. Steps 3 and 4 are repeated until the algorithm termination conditions are achieved.

Algorithm 1: Algorithm for generating macroflows

```
  Input  : Flow data, Macroflow parameters
  Output: Macroflow data
1 Step 1) Summarize flow data
2 Step 2) Generate an initial candidate macroflow set
3 repeat
4     Step 3) Select an operation randomly
5     a) Find a predictable macroflow
6     b) Find an unpredictable macroflow
7     c) Search a new candidate macroflow
8     Step 4) Update a candidate macroflow set
9 until termination conditions;
```

In the first step, the algorithm summarizes observed flow data, that is, selecting top-k attributes in terms of traffic volume for each tuple (srcIP, dstIP, srcPort, dstPort, and proto). Other attribute values are aggregated as "others". This summarization is needed because in the next step, the algorithm generates macroflows from combinations of multiple tuple attribute values (e.g., "dstIP:10.0.0.3", "srcPort:80") and become infeasible if the number of attribute values increases. The summarized flow data are stored in the flow database.

In the second step, the algorithm sets all 1-tuple flows of the summarized flow data, the most coarse-grained flows, as initial candidate macroflows. Our proposed macroflow-generation method adopts a top-down approach that starts from the most coarse-grained flows (1-tuple flows) and divides them into fine-grained ones by specifying other tuple's attribute, which is done in the third step. In this step, the algorithm moves a macroflow from the candidate macroflow set to a predictable or unpredictable macroflow set, as well as generates a new candidate macroflow from candidate macroflow set. The algorithm randomly selects one of the above operations in each iteration. By introducing the random selection, the algorithm can adopt a variety of input-flow data.

The first operation involves moving a predictable macroflow from candidate macroflows. The predictability of macroflow f is measured with its CV in a time series traffic volume x_f. Here, total measurement period is divided into N time slots, and CV_f, which is CV of a flow f, is expressed as:

$$CV_f = \frac{1}{\bar{x_f}}\sqrt{\sum_{t=1}^{N}(x_{f,t} - \bar{x_f})^2} \qquad (1)$$

where $x_{f,t}$ is the traffic volume of the t-th time slot and $\bar{x_f}$ is the average value of the traffic volume of N time slots. When CV_f is lower than a configured threshold value, f is selected as a predictable macroflow. The operation takes a flow f whose traffic volume is the largest among flows whose CV_f are lower than a configured threshold value, and moves f to the predictable macroflow set. The second operation involves moving an unpredictable macroflow whose traffic volume is the largest among flows whose CV_f are higher than the configured threshold value for an unpredictable macroflow. By this elimination, we can expect that the remaining candidates will be more predictable.

The third operation involves generating a new candidate macroflow by combining multiple major attribute values. Specifically, the operation randomly selects a candidate macroflow, and then selects new attribute values that include many 5-tuple flows. After that, it generates a new candidate macroflow by combining the new attribute values and header information of the candidate macroflow.

In the fourth step, the algorithm updates the candidate macroflows. To avoid counting flow data twice, after a macroflow is selected, the remaining candidate macroflow data are reconstructed with flow data except for the selected flow.

After all the steps, the algorithm checks the termination conditions. The conditions, which are given as the input parameters, determine the maximum computation time, maximum repeat count, and residual flow-data-size threshold of flow data for which predictable/unpredictable macroflows have not been classified. The algorithm stops if any of the above conditions are satisfied. Otherwise, it returns to step three. Finally, a macroflow table is generated with all selected macroflows by each OD pair. The priority value of each flow entry is configured in descending order in accordance with the order in which macroflows are selected. All the unclassified flows at the time of finishing the algorithm are classified as unpredictable macroflows with a default match entry composed only of wildcards.

4.3 Route Optimization Methods

Our macroflow-based TE scheme achieves efficient routing by applying an appropriate route-optimization method in accordance with traffic predictability. We incorporated two representative route-optimization methods into our TE scheme. The first method is prediction-based routing, which calculates a route that minimizes the network cost based on the predicted traffic volume. The second method is load-balanced routing, which calculates the route that distributes traffic to multiple links according to the remaining bandwidth of the network.

4.3.1 Prediction-Based Routing

This method computes a routing that minimizes the maximum link utilization of a network for the predicted traffic demand. We adopted a current value-prediction method as a time-series prediction method that directly uses the observed traffic demand of the current time slot, which is used as the predicted traffic demand of the next time slot. The optimized routing of macroflows is determined by solving the optimization problem called the multi-commodity flow problem based on linear programming (MCF-LP) [4], [17]. Since the MCF-LP outputs the multipath routes as the optimal solution, it can be applied only when target flows are splittable. However, the MCF that outputs the single path routes as the optimal solution is NP-hard, whereas the MCF-LP can be solved in polynomial time. The route-optimization problem for prediction-based routing has been formulated as follows.

$$minimize : U \tag{2}$$

$$s.t. : 0 \leq x_{ij}^{pq} \leq 1 \quad (\forall(p,q) \in E, \forall p,q \in N) \tag{3}$$

$$0 \leq U \leq 1 \tag{4}$$

$$\sum_{j:(i,j)\in E} x_{ij}^{pq} - \sum_{j:(i,j)\in E} x_{ji}^{pq} = 0 \quad (\forall p,q \in N, i \neq p, i \neq q) \tag{5}$$

$$\sum_{j:(i,j)\in E} x_{ij}^{pq} - \sum_{j:(i,j)\in E} x_{ji}^{pq} = 1 \quad (\forall p,q \in N, i = p) \tag{6}$$

$$t_{pq} x_{ij}^{pq} \leq U c_{ij} \quad (\forall(i,j) \in E, \forall p,q \in N) \tag{7}$$

Here, V is the set of nodes in a network graph $G(V,E)$, with E being the set of undirected graphs. The link between nodes i and j is denoted as $(i,j) \in E$, and the capacity of the link is denoted as c_{ij}. The terms $p,q \in N$ are the origin and destination nodes of OD traffic demand (p,q), and t_{pq} is the traffic load of each time slot of OD traffic demand (p,q). Let X be the routing matrix that is the set of routing information for all OD flows. The routing variable $x_{ij}^{pq} \in X$, which takes a real value between 0 and 1, is the proportion of passed OD traffic demand (p,q) on link (i,j). The U in Eq. (2) denotes the maximum link utilization in a network. This problem is to seek a routing matrix X to minimize U satisfying the constraints in Eqs. (3)–(7). To obtain a routing matrix of the next time slot, we solve the above problem by using the predictive traffic demand at each time slot.

4.3.2 Load-Balanced Routing

This method does not use any traffic volume information but only residual network capacity to avoid defective routing that induces heavy congestion due to prediction error. The routing of unpredictable macroflows is determined by solving the following linear programming problem, which broadly distributes unpredictable traffic into multiple links to avoid concentrating traffic spikes on any link [8]. Specifically, it calculates multipath routes of each OD flow to minimize the maximum proportion of a passed fraction of a macroflow against the available link capacity in a residual network. The route-optimization problem for load-balanced routing has been formulated as follows.

$$minimize : \sum_{p,q\in Q} h^{pq} \tag{8}$$

$$s.t. : 0 \leq x_{ij}^{pq} \leq 1 \quad (\forall(i,j) \in E, \forall p,q \in Q) \tag{9}$$

$$0 \leq h^{pq} \; (\forall p,q \in Q) \tag{10}$$

$$\sum_{j:(i,j)\in E} x_{ij}^{pq} - \sum_{j:(i,j)\in E} x_{ji}^{pq} = 0$$

$$(\forall p, q \in Q, i \neq p, i \neq q) \tag{11}$$

$$\sum_{j:(i,j)\in E} x_{ij}^{pq} - \sum_{j:(i,j)\in E} x_{ji}^{pq} = 1$$

$$(\forall p, q \in Q, i = p) \tag{12}$$

$$x_{ij}^{pq} \leq h^{pq} c_{ij}$$

$$(\forall (i,j) \in E, \forall p, q \in Q) \tag{13}$$

The value h^{pq} in Eq. (8) denotes the maximum ratio of the OD flow between nodes p and q to the link capacity, which indicates the degree of concentration of traffic flows to the most congested link in the network. When h^{pq} is low, the OD flow between nodes p and q is distributed to multiple links. This problem is to seek a routing matrix X to minimize the sum of h^{pq} satisfying the constraints in Eqs. (9)–(13). This problem is a type of linear programming, and it is much easier to solve than MCF-LP because it does not include traffic volume information; therefore, it can also be solved in polynomial time.

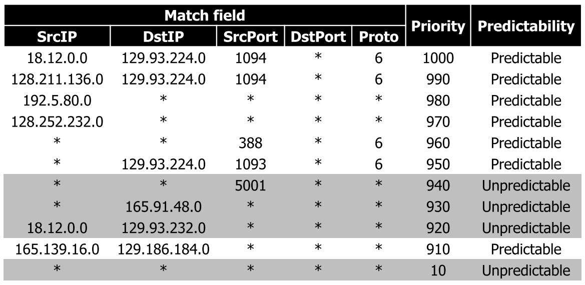
5. Evaluation

In this section, we explain the properties of macroflows generated with our proposed macroflow-generation method. Moreover, we describe the TE performance with our generated macroflows through simulation experiments.

5.1 Dataset

In our evaluation, we used actual NetFlow data and topology information of Internet2 [9]. Internet2 consisted of nine nodes and 26 one-directional links. The OD flows arose between each of the 72 OD node pairs. The NetFlow data including 5-tuple information were exported every five minutes and their source/destination IP addresses were masked by zeroing the last 11 bits of the IP address (addresses are aggregated into 21 prefix). We aggregated NetFlow data into OD flows every five minutes for four hours. That is, four-hour flow data is divided into $N = 48$ time slots, and the length of each time slot is five-minutes. Each OD flow for four hours included hundreds of thousands of 5-tuple flows.

Note that the appropriate time-slot length depends on both application mix in traffic data and requirement for the responsiveness of the TE control-loop. If we set the time-slot length longer, we may miss to capture short spike generated by IoT applications in CV calculation. In addition, because controlling such spike will be done in the next time slot, responsiveness to such traffic spike will decrease. On the other hand, if we set the time-slot length more shorter, longer traffic pattern caused by video distribution will not be captured. In addition, if the slot length and time-scale of the TE control-loop is too short, it leads system overload and frequent fluctuation in traffic route, which makes system unstable and QoE will degrade. Thus, there is trade-off between responsiveness and stability. Based on the above consideration, we set five minutes for the time-slot length as a temporal setting. Pursuing optimal length remains as a future work.



Match field					Priority	Predictability
SrcIP	DstIP	SrcPort	DstPort	Proto		
18.12.0.0	129.93.224.0	1094	*	6	1000	Predictable
128.211.136.0	129.93.224.0	1094	*	6	990	Predictable
192.5.80.0	*	*	*	*	980	Predictable
128.252.232.0	*	*	*	*	970	Predictable
*	*	388	*	6	960	Predictable
*	129.93.224.0	1093	*	6	950	Predictable
*	*	5001	*	*	940	Unpredictable
*	165.91.48.0	*	*	*	930	Unpredictable
18.12.0.0	129.93.232.0	*	*	*	920	Unpredictable
165.139.16.0	129.186.184.0	*	*	*	910	Predictable
*	*	*	*	*	10	Unpredictable

Fig. 4 Example of flow entries to realize predictable and unpredictable macroflows from Chicago to Kansas City.

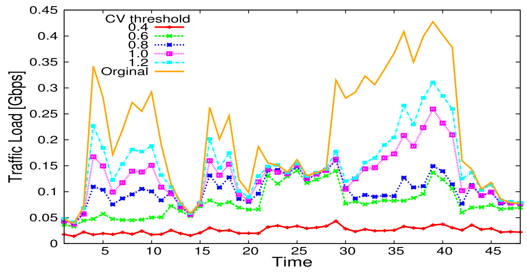


Fig. 5 Traffic variation of predictable macroflow set from Chicago to Kansas City.

5.2 Macroflow Properties

We clarified the macroflow properties generated with our proposed method. We applied it to each piece of NetFlow data aggregated into each OD pair. Our method generated several tens of flow entries for each OD pair. Each macroflow was classified as predictable or unpredictable in accordance with the predictability measured using the CV of its time variation of traffic.

Figure 4 shows an example of a macroflow-classification table generated by analyzing an OD flow from Chicago to Kansas City. Each flow entry has three types of information: "match field", "priority", and "predictability". Flow entries match packets in priority order. This figure shows that our macroflow-generation method succeeded in generating complex combinations of 5-tuple header information. The predictable and unpredictable macroflows were composed of multiple flow entries. A predictability field indicates whether matched traffic is predictable. In a real OpenFlow switch, a predictability field is implemented as the "next-table-id" of "Goto-Table action" [3]. We now discuss our evaluation of the properties of a macroflow set composed of multiple macroflows.

Figure 5 shows the traffic variation of an OD flow and its predictable macroflow set for different CV threshold values for predictable macroflows. The traffic variation of the predictable macroflow set stabilized and the average size of this set decreased as the CV threshold decreased. This indicates a trade-off between the CV and size in a predictable macroflow set.

Figure 6 shows the CV in traffic variation of a pre-

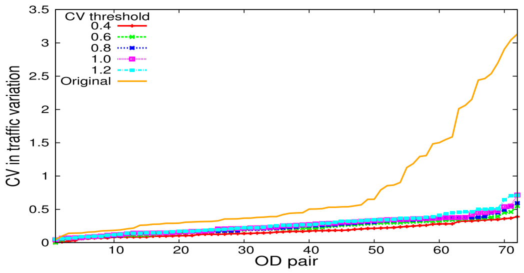


Fig. 6 CV in traffic variation of predictable macroflow set by each OD pair.

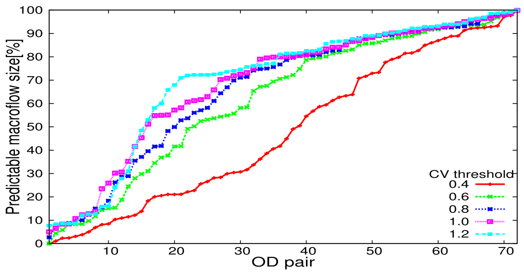


Fig. 7 Size of predictable macroflow set by each OD pair.

dictable macroflow set for each OD pair. Each data point is arranged in ascending order of CV. The CV decreased as the CV threshold decreased. Moreover, the CV in traffic variation of a predictable macroflow set for any CV parameter was much lower than that of the original OD flow. We also evaluated the prediction error of a predictable macroflow when the CV threshold parameter was set to 0.6 and the current value prediction was used. The prediction error was measured using the root mean squared error (RMSE). The results showed that the average prediction error of predictable macroflows was 0.011; in contrast, the average prediction error of OD flows was 0.020. When the linear-prediction method was used, the trend in the results was almost the same. Hence, these results suggest the possibility that our generated macroflows are easier to predict and control than the original OD flows.

Figure 7 compares the size of a predictable macroflow set to that of the OD flow. Each data point is arranged in ascending order of size. The average size of the predictable macroflow set increased as the CV threshold increased. However, the size of the predictable macroflow set for some OD pairs was small in the case of a high CV threshold. This is because the traffic variation of these OD flows drastically changed, so our macroflow-generation method could not extract predictable portions of traffic from them.

Figure 8 plots the number of flow entries in a macroflow-classification table for each OD pair. Each data point is arranged in ascending order of the number of flow entries. The number of flow entries changed little as the CV threshold changed. A macroflow-classification table for each OD pair required an average of 10 flow entries, even though there was an average of about 120,000 microflows within the original OD flows. Moreover, our proposed method took 73 seconds

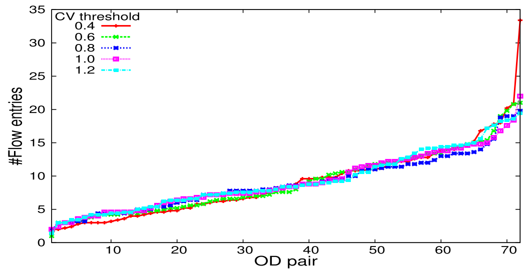


Fig. 8 Number of flow entries to realize predictable and unpredictable macroflows by each OD pair.

on average to generate a macroflow set from 4 hours of OD flow data.

5.3 Traffic Engineering Using Macroflows

We conducted trace-driven simulations to evaluate TE performance by using our generated macroflows. NetFlow data and topology information of Internet2 used in the simulations were the same as in the previous subsection. we used the maximum link load and average link load of the network as evaluation metrics. The CV threshold parameter of our macroflow-generation method was set to 0.6 in the simulations. Our simulations were conducted on a 3.40-GHz core machine with 16-GB of RAM. To solve the optimization problems for TE, we used the GNU Linear Programming Kit mathematical programming software. Its computational time at each time slot was kept within 1 second in our evaluation because the optimization problem was LP and the size of the optimization problem was not so large.

Figure 9 plots the maximum link load of the network at each time slot with the proposed macroflow-based, prediction-based, and load-balanced TE schemes. Here, the prediction-based TE scheme controls all flows by the prediction-based routing in Sect. 4.3.1, and the load-balanced TE scheme controls all flows by the load-balanced routing in Sect. 4.3.2. In the case of the prediction-based TE scheme, the maximum link load of the network increased drastically at times 24 and 44. This is because unexpected traffic variations caused prediction errors; as a result, traffic was concentrated on one link.

Figure 10 shows the time variation of OD traffic originating in Kansas City and destined for Los Angeles. The time variation of this OD traffic changed drastically in a number of time slots including time slots 24 and 44; therefore, it was difficult to predict. The prediction-based TE scheme failed to predict the time variation of this OD traffic accurately, so controlled traffic on the basis of the mis-predicted traffic volume. Consequently, one link was congested because of concentrated traffic. The proposed macroflow-based and load-balanced TE schemes distributed such drastic traffic variations to a number of links. Consequently, the proposed scheme avoided a drastic increase in the maximum link load, even at time slots 24 and 44 and reduced the maximum link load by 34% at time slot 24 compared to prediction-based TE scheme. This reduction was also achieved by the load-

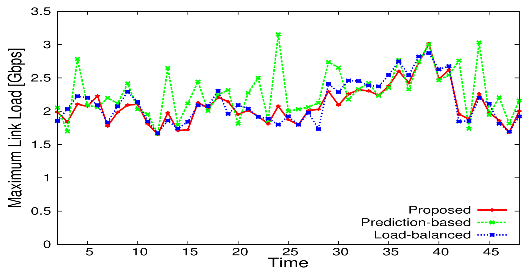


Fig. 9 Maximum link load in network for each time slot.

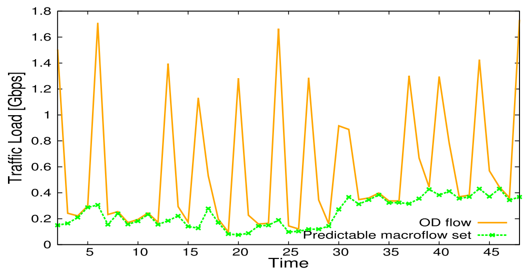


Fig. 10 Time variation of OD traffic from Kansas City to Los Angeles.

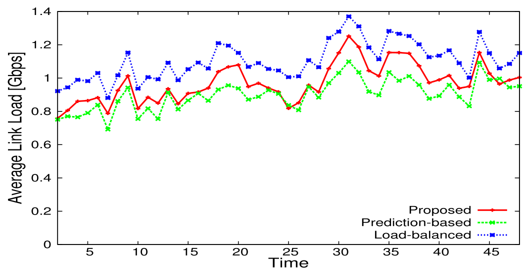


Fig. 11 Average link load in network for each time slot.

balanced TE scheme.

Figure 11 shows the average link load of the network at each time slot with the proposed macroflow-based, prediction-based, and load-balanced TE schemes. This figure illustrates that the average link load of the network with the prediction-based TE scheme was lower than that of the load-balanced TE scheme. The average length of paths set by the prediction-based TE scheme was close to the length of the shortest paths. The load-balanced TE scheme set indirect paths on all OD flows, and their average length increased. Longer paths result in greater consumption of network resources; consequently, a higher average link load. Because our proposed scheme sets indirect paths on only unpredictable macroflows, it reduced the average link load of the network by 11% compared with the load-balanced TE scheme that sets indirect paths on all OD flows.

These results show that our proposed macroflow-based TE scheme prevents increases in the maximum link load caused by drastic traffic variation and the average link load caused by an increase in path lengths.

6. Discussion

6.1 Realization Issues

We discuss about the issues for realizing our proposed TE on an SDN/OpenFlow network.

Our proposed macroflow-based TE scheme requires frequent flow-table updates. However, while a flow table is being updated to install new routes, OpenFlow switches cannot handle traffic appropriately. A flow table will remain incomplete in the meantime, which impairs network robustness due to traffic classification or forwarding error. In our verification experiment, to reduce the time for a flow table to remain incomplete, Our flow-table architecture prepares two types of tables: active and standby. First, all new routes are installed into a standby table. After a standby flow table is updated, an OpenFlow switch changes an active flow table from the current one to a new one. Specifically, we change the value of "next-table-id" in hierarchical flow table to shift over to a new flow table. However, it is difficult for common physical OpenFlow switches to maintain flow-table space for standby because they have a limitation regarding the number of flow tables or TCAM volume. In networks in which physical OpenFlow switches work, a scheme to ensure network consistency while flow tables are being updated is necessary to conduct dynamic TE.

6.2 Improvement of Methods to Enable Macroflow-based TE Scheme

In this paper, we adopt simple methods for traffic prediction and predictable flow detection; current value-prediction method and CV-based detection method. We do not claim that these methods are the most suitable ones to our macroflow-based traffic engineering. Rather, we adopt these methods in order to clarify the effectiveness of the macroflow-based traffic engineering scheme even with such naive methods. Of course, by adopting more sophisticated methods, it is possible that performance of traffic engineering will be improved. In this subsection, we discuss these possibilities.

As for the traffic prediction, there is a vast amount of literature for time-series prediction methods and its application to traffic prediction. Representative one is a method that predicts the future value with a linear combination of past observation and noise term, such as ARIMA [19]. Recently, traffic prediction models with more complicated structure by using Deep Neural Networks such as LTSM has been proposed [20]. These methods can improve the performance of prediction-based TE scheme and thus, they also improve the performance of our proposed scheme that is the combination of prediction-based and load-balanced. Though it is true that performance advantage of our scheme to prediction-based only TE scheme may decrease with such sophisticated prediction methods, there still remains the risk such methods fails to predict sudden traffic spike as shown in Fig. 10. Even in such case, our macroflow-based TE scheme can mitigate them with load-balance routing. Thus we expect that the advantage of our scheme to the prediction-based method still remains.

As for the predictable flow detection, if we change the measure of the unpredictability from CV of a flow to Kullback-Leibler divergence between the predicted distribution of flow size and actual distribution of flow size, the predictability of traffic can be more accurately measured. Though with naive CV-based predictable flow detection methods generate fairly accurate predictable macroflow as in Fig. 5, such sophisticated predictable flow detection methods can more accurately generate predictable/unpredictable macroflow set. Adopting such sophisticated methods remains as a future work.

7. Conclusion

We proposed a macroflow-based TE scheme that applies the most appropriate routing policy to each macroflow in accordance with the predictability of its traffic rate. We also proposed a method for identifying predictable and unpredictable macroflows from flow data in accordance with the degree of traffic variation. Through simulations conducted with actual traffic traces, we demonstrated that the proposed scheme can both optimize the performance of predictable traffic and enable robust control for unpredictable traffic. For future work, we plan to evaluate the performance of our macroflow-based TE scheme by synthesizing different types of routing algorithms and prediction methods in accordance with other traffic characteristics on a large-scale OpenFlow testbed network.

Acknowledgments

This work was supported in part by the Strategic Information and Communications R&D Promotion Programme (SCOPE) of the Ministry of Internal Affairs and Communications, Japan.

References

[1] Y. Takahashi, K. Ishibashi, M. Tsujino, N. Kamiyama, K. Shiomoto, T. Otoshi, Y. Ohsita, and M. Murata, "Separating predictable and unpredictable flows via dynamic flow mining for effective traffic engineering," Proc. IEEE ICC, July 2016.

[2] 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, April 2008.

[3] "OpenFlow switch specification v1.4.0," Oct. 2013.

[4] N. Wang, K. Ho, G. Pavlou, and M. Howarth, "An overview of routing optimization for internet traffic engineering," IEEE Commun. Surveys Tuts., vol.10, no.1, pp.36–56, April 2008.

[5] T. Otoshi, Y. Ohsita, M. Murata, Y. Takahashi, K. Ishibashi, and K. Shiomoto, "Traffic prediction for dynamic traffic engineering," Computer Networks, vol.85, pp.36–50, July 2015.

[6] C. Katris and S. Daskalaki, "Comparing forecasting approaches for Internet traffic," Expert Systems with Applications, vol.42, no.21, pp.8172–8183, Nov. 2015.

[7] D. Applegate and E. Cohen, "Making intra-domain routing robust to changing and uncertain traffic demands: Understanding fundamental tradeoffs," Proc. ACM SIGCOMM, pp.313–324, Aug. 2003.

[8] M. Antic, N. Maksic, P. Knezevic, and A. Smiljanic, "Two phase load balanced routing using OSPF," IEEE J. Sel. Areas. Commun., vol.28, no.1, pp.51–59, Jan. 2010.

[9] "Internet2 data," available from http://internet2.edu/observatory/archive/data-collections.html

[10] Y. Hu, D.-M. Chiu, and J.C.S. Lui, "Entropy based adaptive flow aggregation," IEEE/ACM Trans. Netw., vol.17, no.3, pp.698–711, June 2009.

[11] Y. Zhang, S. Singh, S. Sen, N. Duffield, and C. Lund, "Online identification of hierarchical heavy hitters," Proc. ACM IMC, pp.101–114, Oct. 2004.

[12] N. Kamiyama, Y. Takahashi, K. Ishibashi, K. Shiomoto, T. Otoshi, Y. Ohsita, and M. Murata, "Flow aggregation for traffic engineering," Proc. IEEE GLOBECOM, pp.1936–1941, Dec. 2014.

[13] B. Fortz, J. Rexford, and M. Thorup, "Traffic engineering with traditional IP routing protocols," IEEE Commun. Mag., vol.40, no.10, pp.118–124, Oct. 2002.

[14] A. Elwalid, C. Jin, S. Low, and I. Widjaja, "MATE: MPLS adaptive traffic engineering," IEEE INFOCOM2001, 2001.

[15] S. Jain, M. Zhu, J. Zolla, U. Hölzle, S. Stuart, A. Vahdat, A. Kumar, S. Mandal, J. Ong, L. Poutievski, A. Singh, S. Venkata, J. Wanderer, and J. Zhou, "B4: Experience with a globally deployed software defined WAN," Proc. ACM SIGCOMM, pp.3–14, Aug. 2013.

[16] C.-Y. Hong, S. Kandula, R. Mahajan, M. Zhang, V. Gill, M. Nanduri, and R. Wattenhofer, "Achieving high utilization with software-driven WAN," Proc. ACM SIGCOMM, pp.15–26, Aug. 2013.

[17] D. Mitra and K.G. Ramakrishnan, "A case study of multiservice, multipriority traffic engineering design for data networks," Proc. IEEE GLOBECOM, pp.1077–1083, Dec. 1999.

[18] B. Pfaff, et al., "Extending networking into the virtualization layer," Proc. ACM SIGCOMM Workshop on Hot Topics in Networking, Oct. 2009.

[19] P.J. Brockwell and R.A. Davis, Introduction to Time Series and Forecasting, Springer, 2002.

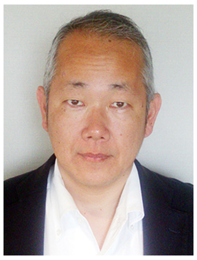
[20] F.A. Gers, D. Eck, and J. Schmidhuber, "Applying LSTM to time series predictable through time-window approaches," Proc. Neural Nets WIRN Vietri-01, pp.193–200, Springer, May 2001.

Yousuke Takahashi received his B.S. and M.S. in information science from Osaka University in 2007 and 2009. He joined NTT Laboratories in 2009 and has been engaged in researches on network management and traffic engineering. From 2016, he has been engaged in research on network management in NTT Communications Corporations. He is a member of IEICE.

Keisuke Ishibashi received his B.S. and M.S. in mathematics from Tohoku University in 1993 and 1995, and received his Ph.D. in information science and technology from the University of Tokyo in 2005. Since joining NTT in 1995, he has been engaged in research on traffic and performance issues in computer communication networks. He is a member of the IEEE and the Operations Research Society of Japan.

Masayuki Tsujino received the B.E., and M.E., degrees in applied mathematics and physics from Kyoto University, Kyoto in 1990 and 1992, respectively. He joined the Nippon Telegraph and Telephone Corporation (NTT), Tokyo, Japan in April 1992. From February 1999 to March 2005, he was engaged in NTT Communication Corporation. He has been engaged in R&D of access network design, IP traffic management and traffic engineering technologies. He is a member of IEICE and the Operations Research Society of Japan.

Noriaki Kamiyama received his M.E. and Ph.D. degrees in communications engineering from Osaka University in 1994 and 1996, respectively. From 1996 to 1997, he was with the University of Southern California as a visiting researcher. He joined NTT Multimedia Network Laboratories in 1997, and he has been at NTT Network Technology Laboratories by 2016. He was also with the Osaka University as an invited associate professor from 2013 to 2014 and an invited professor in 2015. From 2017, he is a professor of Fukuoka University. He has been engaged in research concerning content distribution systems, network design, network economics, traffic measurement and analysis, traffic engineering, and optical networks. He received the best paper award at the IFIP/IEEE IM 2013. He is a member of IEEE and IEICE.



Kohei Shiomoto is a Professor of Tokyo City University, Tokyo Japan. His current interest research areas include software-defined networking, network function virtualization, machine-learning, and network management. From 1989 to 2017, in NTT Laboratories, he was engaged in research and development of high-speed networks including ATM networks, IP/MPLS networks, GMPLS networks, network virtualization, traffic management, network analytics. From 1996 to 1997 he was engaged in research in high-speed networking as a visiting scholar at Washington University in St. Louis, MO, USA. He received his B.E., M.E., and Ph.D. degrees in information and computer sciences from Osaka University, Osaka in 1987 1989, and 1998. He is a Fellow of IEICE, a Senior Member of IEEE, and a member of ACM.

Tatsuya Otoshi received an M.E. and Ph.D. degrees in information science and technology in 2014 and 2017 from Osaka University, where he is currently a specially appointed assistant professor in the Graduate School of Information Science and Technology. His research interests include traffic engineering and traffic prediction. He is a member of IEICE and IEEE.

Yuichi Ohsita received M.E. and Ph.D. degrees in information science and technology in 2005 and 2008 from Osaka University, where he is currently an assistant professor in the Graduate School of Information Science and Technology. His research interests include traffic matrix estimation and countermeasures against DDoS attacks. He is a member of IEICE, IEEE, and the Association for Computing Machinery (ACM).

Masayuki Murata received M.E. and Ph.D. degrees in information science and technology from Osaka University in 1984 and 1988. In April 1984, he joined the Tokyo Research Laboratory IBM Japan as a researcher. From September 1987 to January 1989, he was an assistant professor with the Computation Center, Osaka University. In February 1989, he moved to the Department of Information and Computer Sciences, Faculty of Engineering Science, Osaka University. From 1992 to 1999, he was an associate professor with the Graduate School of Engineering Science, Osaka University, and since April 1999, he has been a professor. He moved to the Graduate School of Information Science and Technology, Osaka University in April 2004. He has published more than 300 papers in international and domestic journals and conferences. His research interests include computer communication networks, performance modeling, and evaluation. He is a fellow of IEICE and a member of IEEE, the Association for Computing Machinery (ACM), The Internet Society, and IPSJ.