Traffic Engineering Based on Stochastic Model Predictive Control for Uncertain Traffic Change

Tatsuya Otoshi†, Yuichi Ohsita†, Masayuki Murata†,
Yousuke Takahashi††, Keisuke Ishibashi††, Kohei Shiomoto††, and Tomoaki Hashimoto‡
†Graduate School of Information Science and Technology, Osaka University
1-5 Yamadaoka, Suita, Osaka 565-0871, Japan
Email: {t-otoshi, y-ohsita, murata}@ist.osaka-u.ac.jp
††NTT Network Technology Laboratories, NTT Corporation
3-9-11, Midori-Cho Musashino–Shi, Tokyo 180-8585 Japan
Email: {takahashi.yousuke, ishibashi.keisuke, shiomoto.kohei}@lab.ntt.co.jp
‡Graduate School of Engineering Science, Osaka University
1-3 Machikaneyama-Cho, Toyonaka, Osaka 560-8631, Japan
Email: thashi@sys.es.osaka-u.ac.jp

***Abstract*—Traffic engineering (TE) plays an essential role in deciding routes that effectively use network resources. This is particularly important when one considers the increasing time variation of Internet traffic such as streaming and cloud services. Traffic engineering with traffic prediction is one approach to stably accommodating time-varying traffic. This approach calculates routes from predicted traffic to avoid congestion, but predictions may include errors that instead cause congestion. We propose a prediction-based traffic engineering method that is robust to prediction errors by considering the probability distribution of predicted traffic. Our approach is based on a control-theoretic approach called *stochastic model predictive control*. Routes are calculated using a probability distribution of prediction errors so that the occurrence probability of congestion is lower than an operator-specified level. By considering the multi-step future dynamics of traffic, the routes are changed gradually to avoid route oscillation. We also show a relaxation method for unreliable far-future probabilistic constraints to avoid overly conservative route changes. Through simulations using backbone network traffic traces, we demonstrate that our method can accommodate most traffic variations under a given target link capacity without sudden large routes changes.**

***Index Terms*—Stochastic Model Predictive Control, Traffic Engineering, Traffic Prediction, Multi-path Routing**

I. Introduction

Traffic engineering (TE) plays an essential role in deciding routes that effectively use network resources. This is particularly important when one considers increasing time variation of Internet traffic such as streaming and cloud services. In the past, backbone networks have addressed this problem by reserving redundant link capacity to accommodate traffic surges [1]. However, doing so incurs significantly higher costs as the average and variance of traffic increases; poor network resource utility tends to reserve more than double the capacity required to accommodate the actual traffic. Dynamic TE methods have been studied for treating time-varying traffic in a way that effectively utilizes limited resources [2–4]. In the dynamic TE method, a control server periodically observes network traffic and dynamically reroutes flows to accommodate the observed traffic. However, such methods set routes for only observed traffic. This renders configured routes unsuitable if traffic changes happen, because routes are not changed during the next control cycle. One might think that the control server should quickly respond to traffic changes by shortening the control cycle interval, but doing so can induce frequent route changes resulting in route oscillation that degrades TCP session throughput; oscillations cause packet reordering by delivering the packets of a given TCP session via different paths, which reduces the TCP session window size. Route oscillations also cause overly frequent changes in round-trip time (RTT), which decreases the throughput of delay-based TCP [5]. Hence, a dynamic TE that avoids congestion without significant route changes is required.

One approach to solving such problems is to use the predicted traffic. In [6], we have presented such a TE method, which is based on a control theory of predictive control called *model predictive control (MPC)*. In this method, the control server calculates the routes of several future time slots (or control cycles), considering predicted traffic variation. The control server then sets up only the next-step routes, and corrects the traffic prediction by reflecting traffic changes newly observed. It can then follow traffic variation without significantly changing the routes. Of course, traffic prediction errors cannot be wholly avoided even in the above method, and prediction errors may cause congestion in the next step.

In this paper, we discuss the TE method that is robust to prediction errors. One approach to handling prediction uncertainty is to utilize a probability distribution of prediction errors. Fortunately, MPC can consider the probability distribution, by introducing the constraints that the risk of control failure should be less than a predefined threshold. MPC considering the probabilistic distribution is called *stochastic MPC(SMPC)* [7].

In this paper, we apply SMPC to TE. We call this *stochastic model predictive traffic engineering (SMP-TE)*. We first model the problem of TE as an optimization problem that maximizes the network performance under the constraints that the probability of congestion should be less than a predefined threshold. In SMP-TE, this optimization problem is solved by the controller so as to obtain the routes for future time slots. Then, similar to the TE based on MPC [6], the controller sets up only the next-step routes, and recalculates the routes

for the future time slot after the correction of the traffic prediction by reflecting newly observed traffic data. SMP-TE avoids congestion without requiring a large amount of excess network resources even if prediction causes errors, because the optimization problem considers the distribution of prediction errors.

In SMP-TE, one of the important problems is the threshold for the probability of congestion. Generally, the prediction errors of the traffic in the far future become large. Such a large prediction error may cause the unnecessary route changes. To solve this problem, we also propose a relaxation of probability constraints.

The rest of this paper is organized as follows. Section II explains the overview of TE. Section III describes our TE method, SMP-TE, in which we apply SMPC to the TE method. Section IV presents an evaluation of our TE method comparing with the simple prediction-based TE. Section V presents our concluding remarks.

II. Dynamic Traffic Engineering

TE has been studied as an approach to accommodating changing traffic by dynamically changing routes [2–4]. The basic process of TE is composed of the following three steps: (1) traffic information is obtained, (2) routes are calculated based on the traffic information, and (3) the calculated routes are applied to the actual network. These steps are periodically repeated to follow traffic changes.

At each control interval, a control server collects the traffic rates of each origin–destination (OD) flow that traverses from the ingress point-of-presence (PoP) router to the egress PoP router. After the traffic information is collected, routes are calculated so as to avoid the congestion. The routes are defined by the fraction of traffic of each OD flow sent to each path. Hereafter, we denote the traffic rate of OD flow i at the kth time slot by $x_i(k)$, and the vector $\boldsymbol{x}(k) = {}^t(x_1(k), \cdots, x_F(k))$ represents the traffic rates of all OD flows at the kth time slot, where F is the number of OD flows. Also, we denote the routing fractions by a matrix $R(k)$ whose (i,j) element $R_{i,j}(k)$ indicates the fraction of traffic on the OD flow j that traverses the available path i. The traffic rates on links are calculated as

$$\boldsymbol{y}(t) = G \cdot R(t) \cdot \boldsymbol{x}(t) \tag{1}$$

where $y(t) = (y_1(t), \cdots, y_l(t))$ is a vector where component $y_i(t)$ indicates the expected traffic rate of link i at the time slot t, l is the number of links, and G is a matrix whose (i,j) element $G_{i,j}$ is 1 if the available path j traverses the link i and 0 otherwise.

TE methods set routes $R(t)$ so as to maximize the network performance without causing congestion. The network performance is maximized by minimizing a cost function $f(R(t))$ such as end-to-end delay, hop length and so on. On the other hand, congestion can be avoided by setting $R(t)$ so that $y_l(t) \leq C_l$, where C_l is the capacity of the link l. However, when setting $R(t)$, the actual value of $x(t)$ is not known yet. Most of the TE methods uses $x(t-1)$ instead of $x(t)$ when calculating $R(t)$. However, the calculated routes $R(t)$ are not exactly suited to the traffic rate at time slot t, because the actual traffic rate $x(t)$ is different from $x(t-1)$. Frequent control with narrow time slots is one way to quickly respond to traffic fluctuations, but frequent and significant route changes degrade the throughput of TCP sessions due to the packet reordering or frequent changes in RTT.

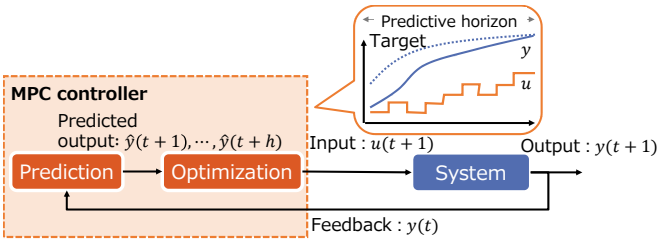


Fig. 1. Overview of MPC

One approach to solving this problem is to use the predicted traffic rate. If we can predict future traffic accurately, the TE using the predicted traffic avoids the future congestion. However, predicted traffic includes prediction errors. Thus, a TE method considering the prediction errors is required.

III. SMP-TE: Stochastic Model Predictive Traffic Engineering

Before describing our SMP-TE, we first briefly introduce SMPC in Subsection III-A. We then move to our proposed SMP-TE in Subsection III-B.

A. Stochastic Model Predictive Control

1) Model Predictive Control: First, we briefly show the concept of MPC. MPC is a method of system control based on predictions of system dynamics that has been studied in recent years. Figure 1 shows an overview of MPC. In MPC, a controller sets an input parameter so as to maintain system performance at close to an operator-specified target. Unlike traditional system control, the MPC controller predicts changes in the output value to calculate inputs for the *predictive horizon*, time slots $[t+1, t+h]$ where h is the distance to the predictive horizon. We denote the input and output at the kth time slot by $u(k)$ and $y(k)$, respectively. The MPC controller calculates the inputs for the predictive horizon $[t+1, t+h]$ so as to keep $y(k)$ close to the target value $r_y(k)$. The inputs $u(t+1), \cdots, u(t+h)$ that keep $y(k)$ close to $r_y(k)$ are obtained using the objective function $J_1 = \sum_{k=t+1}^{t+h} \|y(k) - r_y(k)\|^2$, where $\|\cdot\|$ represents the Euclidean norm:

$$(u(t+1), \cdots, u(t+h)) = \underset{(u(t+1), \cdots, u(t+h))}{\arg\min} J_1. \tag{2}$$

To solve the above optimization problem, future outputs $y(t+1), \cdots, y(t+h)$ must be predicted from inputs $u(t+1), \cdots, u(t+h)$. The future output under a given input is calculated by a system model that represents the system dynamics. In system control, a system model is often represented by a mathematical formula, the *state space representation*, described as

$$z(k+1) = \phi(k, z(k), u(k)) \tag{3}$$

$$y(k) = \psi(k, z(k), u(k)), \tag{4}$$

where $z(k)$ is the state of the system at the kth time slot, and ϕ and ψ are functions that respectively map the current state and input onto the next state and output. We use $\hat{y}(k)$ as the predicted value of $y(k)$.

Modeling the system by a mathematical formula, however, may entail modeling errors. The prediction of system output

will entail error under such an incorrect model, and prediction errors become increasingly large with more distant predictive horizons. One approach to solve this increasing error is to use feedback of actual system output. That is, the MPC controller implements only the first of the calculated inputs $u(t+1)$. Then, the MPC controller observes the output and corrects the prediction, using the output value. After prediction correction, the MPC controller recalculates the input value for the next time slot with the corrected prediction.

Prediction errors may also significantly change input values, destabilizing the system. The controller therefore restricts the amount of allowed change to inputs, which mitigates the influence of prediction errors. We denote the amount of change in the input at the time slot k by $\Delta u(k) = u(k) - u(k-1)$, and the aggregated amount of change during the predictive horizon by $J_2 = \sum_{k=t+1}^{t+h} \|\Delta u(k)\|$. Instead of the input values determined by Eq. (2), the controller calculates the input values by the following optimization problem:

$$(u(t+1), \cdots, u(t+h)) = \underset{(u(t+1),\cdots,u(t+h))}{\arg\min} \quad J_1 + wJ_2, \quad (5)$$

where w is a parameter for weighting the two objective functions J_1 and J_2.

2) Probability Constraints: Realistic systems entail input or output constraints such as physical constraints and boundary conditions. Here, if the system has an upper bound on output, the MPC controller needs to search the optimal input under the constraint

$$y(t+k) \leq y_u, \ k = 1, \cdots, h, \quad (6)$$

where y_u is the upper bound of the output value.

If a modeling error exists, the calculated input may be infeasible, which violates the constraint (6). Given the exact upper bound $\epsilon > y(t+k) - \hat{y}(t+k)$, the controller determines the input without the constraint violation by maintaining the constraint

$$\hat{y}(t+k) + \epsilon \leq y_u, \ k = 1, \cdots, h. \quad (7)$$

However this hard constraint causes overly conservative control, and guaranteeing applicability to the worst-case scenario will considerably degrade control performance. Additionally, exact upper bounds are rarely available in actual situations.

A probability distribution of error can determine a soft bound to use in place of an exact bound on modeling error. Assuming that the k-th ahead modeling error $\epsilon_y(t+k)$ is a random variable following a certain probability distribution, the output value $y(t+k) = \hat{y}(t+k) + \epsilon_y(t+k)$ is also a random variable. Then, the probability that $y(t+k)$ violates the upper bound can be defined, and we denote the probability as $P[y(t+k) > y_u]$. SMPC is a control method that deals with such random variables of output. To avoid constraint violations, the violating probability should be at a certain level p. Then, the controller calculates safe inputs by the probability constraint

$$P[y(t+k) > y_u] < p, \ k = 1, \cdots, h. \quad (8)$$

Eq. (8) becomes a harder constraint when p is small, and Eq. (8) is equivalent to Eq. (7) when $p = 0$. Even allowing for the rare case of constraint violation according to p, the controller can still robustly avoid performance degradation to model errors.

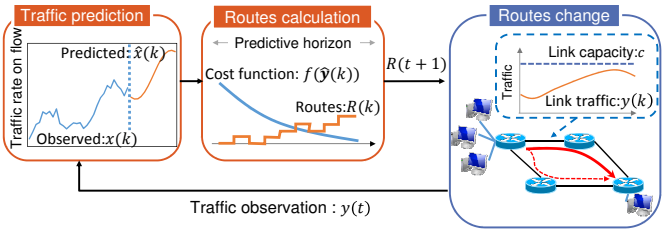


Fig. 2. Overview of SMP-TE

B. Applying SMPC to TE

1) TE Model for SMPC: We apply SMPC to TE, and realize a prediction-based TE that is robust to prediction errors. Figure 2 shows an overview of our TE method, to which SMPC is applied. We assume that a control server collects all traffic information and sets the routes. In the TE, a central control server acts as the MPC controller, which inputs routes $R(k)$ and measures network outputs and the traffic rates on links $\boldsymbol{y}(k)$. The control server periodically changes routes by repeating the following two steps: 1) The control server predicts the traffic rates of OD flows for the target time slots from the previously observed traffic rates. 2) The control server calculates routes from the prediction so as to minimize a cost function $f(R(k))$ while maintaining a low congestion occurrence probability.

2) Formulation of the Optimization Problem: To avoid congestion caused by prediction errors, we use probabilistic constraints as capacity constraints. Given target capacities C_l and probability p, the controller maintains the occurrence probability of capacity overshooting $P[y_l(k) > C_l]$ under p. With this constraint, the control server computes routes by considering objective functions $J_1 = \sum_{k=t+1}^{t+h} f(R(k))$, which indicates a summation of the cost function at each time slot, and $J_2 = \sum_{k=t+1}^{t+h} \|\Delta R(k)\|^2$, which indicates the sum of squares of the amount of route changes. This multi-objective optimization is conducted by minimizing the weighted sum $(1-w)J_1 + wJ_2$, where $0 \leq w \leq 1$ weights the importance of the restriction on route changes.

In SMP-TE, the control server solves the following optimization problem at each time slot t:

$$minimize \sum_{k=t+1}^{t+h} \left((1-w)f(R(k)) + w\|\Delta R(k)\|^2\right) \quad (9)$$

$$subject\ to \ \forall k, \hat{\boldsymbol{y}}(k) = G \cdot R(k) \cdot \hat{\boldsymbol{x}}(k) \quad (10)$$

$$\forall k, l, P[y_l(k) > C_l] \leq p \quad (11)$$

$$\forall k, \forall i, \forall j, R_{i,j}(k) \in [0, 1] \quad (12)$$

$$\forall k, \sum_{i \in \wp(j)} R_{i,j}(k) = 1. \quad (13)$$

where $\hat{\boldsymbol{x}}(k)$ is predicted value of $\boldsymbol{x}(k)$, $wp(j)$ is the set of available paths of OD flow j, and G is a matrix whose (i,j) element $G_{i,j}$ is 1 if the path j traverses the link i and 0 otherwise. Here, $\hat{\boldsymbol{x}}(k), G$ are given variables and $R(k), \hat{\boldsymbol{y}}(k)$ are the variables to be optimized. Eq. (10) represents the relation between the traffic rates of the OD flows and links. Eq. (11) is the probabilistic constraint that the probability of congestion occurrence is lower than p. Eqs. (12) and (13) mean that all traffic on each OD flow is allocated to an available path.

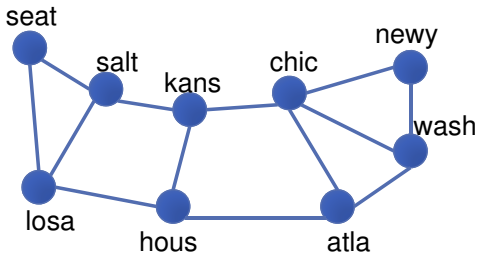


Fig. 3. Internet2 topology

Although all of the routes $R(t+1), \cdots, R(t+h)$ during the predictive horizon are obtained by solving the above optimization problem, the control server implements only the next routes $R(t+1)$. After the route change, the control server corrects the traffic prediction $\hat{\boldsymbol{x}}(k)$ using the newly observed traffic rate, and recalculates the next routes by solving the optimization problem again.

3) Relaxation of Future Probabilistic Constraint: In the above formulation of SMP-TE, the probability p was constant for all time slots within the predictive horizon. However, prediction accuracy for the next time slot is more important in the current model setting. Also, further future prediction is less reliable. Accordingly, forcing the same level of probabilistic constraint for unreliable far-future predictions incurs unnecessary routes changes. This is because the probability $P[y_l(k) > C_l]$ becomes large when the $y_l(k)$ has large variance, and constraints Eq. (11) becomes more active.

To solve this problem, we introduce the increasing probability $p(k)$ for the capacity constraints. By replacing the p in Eq. (11) with $p(k)$, the probabilistic constraint is gradually relaxed as time slots advance. There are many possible approaches to relaxing the probability. In this paper, we exponentially decrease the complement probability $q(k) = 1-p(k)$ as

$$q(k) = (1-p)\exp(-\frac{k-t-1}{\tau}), \tag{14}$$

where τ is the time constant that determines the decreasing speed. If $q(k)$ is less than 0.5, even the expected traffic rates are not accommodated by the calculated routes. In this case, calculated routes no longer avoid the congestion, hence it does not make sense to consider the case of $q(k) < 0.5$. We thus limit the minimum value of $q(k)$ as 0.5.

IV. Evaluation

A. Simulation Environment

1) Network Topology: In the following evaluation, we use the Internet2 backbone network (Fig. 3). The Internet2 backbone network has 9 PoP routers and 13 bidirectional links.

2) Traffic: We use actual traffic traces [8] monitored from 00:00 on 6 Feb 2014 to 23:59 on 12 Feb 2014. These traffic data were collected by the Netflow protocol at each of the PoP routers. The sampling rate is one out of every 100 packets, and aggregated data are exported every 5 min. Though sampled data may contain sampling errors, those errors do not have a large impact on our evaluation because a huge number of flows are aggregated into OD flows, in which the aggregated error of each flow is much smaller than the total traffic amounts.

Our interest lies on how the SMP-TE can avoid the congestion under limited resources and the existence of prediction error. However, in fact, the Internet2 network is not congested due to an over-provisioning of link capacity; the maximum link utilization is less than 20%. Accodingly, we artifically set up a congested environment by multiplying actual traffic amounts by 5, and setting the target link utilization to 95% in the following evaluation. The traffic data used in our evaluation is shown in Fig. 4, where the time slot length is set to 2 h.

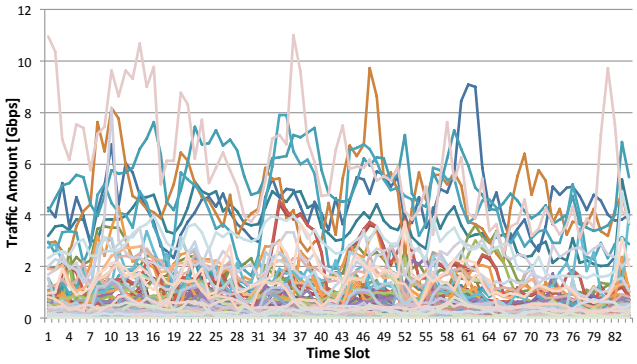


Fig. 4. Internet2 network traffic (time slot granularity: 2 h)

3) Prediction Error Model: In our evaluation, the predicted traffic rates are given by adding prediction errors to the actual traffic rates to evaluate the SMP-TE without impact of the specific prediction model. The prediction error is generated so as to follow a zero-mean Gaussian distribution, based on the existing papers on prediction methods [9, 10].

Assuming that the prediction error of each time slot is independent, the variance of prediction error in the kth ahead time slot is $\sigma^2 k$ where σ^2 is the variance of one-step prediction error. We set the variance of one-step prediction error on flow j based on a normalized prediction error metric called *normalized mean squared error (NMSE)*:

$$\text{NMSE} = \frac{\sigma_j^2}{V[x_j(t)]}, \tag{15}$$

where σ_j^2 is the variance of prediction error on flow j, and $V[x_j(t)]$ is the variance of actual traffic. Therefore, we set σ_j^2 to $0.3V[x_j(t)]$ based on typical NMSE values [11].

4) Cost Function: In our evaluation we use the average hop length as a cost function, because reducing hop length lowers propagation delay in the calculated routes. Because the queuing delay is negligible when the link load is under a certain targeted capacity, we minimize the end-to-end delay by minimizing the propagation delay. The average hop length D is defined using R; $D = \frac{1}{F}\sum_j \sum_{i \in \wp(j)} R_{i,j} d_i$, where d_i is the path length of i. We use the normalized hop length $\frac{D}{\max_j d_j}$ as a cost function. Though we conduct the simulation with changing the weighting parameter w from 0 to 1 by 0.1, we show only the result with $w = 0.5$ because the similar result is obtained with any $0 < w < 1$.

5) Routing Calculation: To solve the probabilistic optimization Eqs. (9)–(13), we transform the probabilistic constraint Eq. (11) into a deterministic constraint. Similaly to [12], the probabilistic constraint Eq. (11) is equally replaced by following deterministic constraint

$$\forall k, \forall l, \hat{y}_l(k) + \Phi^{-1}(1-p)\sqrt{\sum_j A_{l,j}(k)^2 \sigma_j^2 k} \le C_l(k), \tag{16}$$

where Φ^{-1} is the quantile function of the Gaussian distribution, and $A_{i,j}(k)$ is the (i,j)-element of the matrix $A(k) = G \cdot R(k)$, which indicates the fraction of OD flow j traversing link i.

As a result of the transformation, our optimization problem Eqs. (9)–(13) is equally replaced by a convex optimization program called *second-order cone programming (SOCP)*. We use the optimization problem solver CPLEX [13] package to solve the SOCP. We ran CPLEX on a computer with four Intel Xeon E7-4870 processors. The calculation of each time slot is finished within a few seconds for Internet2 topology.

6) Compared Methods: We compare our SMP-TE method with two prediction-based TE methods. The first is the simplest TE method, which uses only one-step ahead prediction without considering the probability distribution of prediction error. This is a special case of our method when parameters are set to $h = 1$ and $p = 0.5$. Comparison with this method demonstrates the effect of multi-step prediction.

The second method is simple MPC-based TE similar to [6], which uses multi-step-ahead prediction without considering the probability distribution. This is also a special case of SMP-TE with parameter setting $p = 0.5$. Hereafter, we call this MPC-based TE as MP-TE. Comparison with this method confirms that the stochastic constraint is effective for avoiding the impact of prediction error without causing significant route changes.

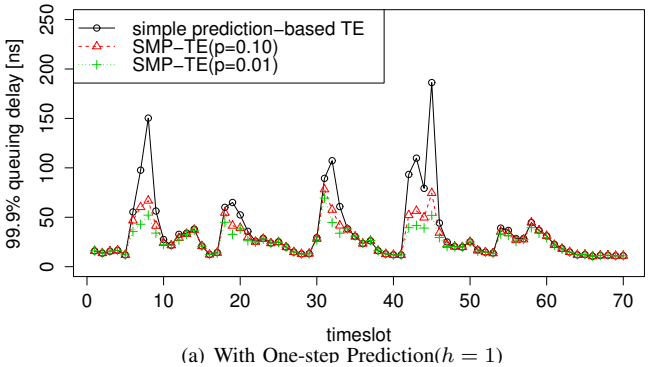
B. Effect of Stochastic Constraint

First, we show how the stochastic constraint is effective for avoiding congestion caused by prediction error. Figure 5 shows the queuing delay of the bottleneck link. Shown is a 99.9% delay, which means that 99.9% of packets will experience delay caused by queuing on a link lower than this value. We calculated the link delay from link utilization by approximating packet processing on the Internet by the M/M/1 queuing model. According to queuing theory, 99.9% delay is calculated as $-\log(1-0.999)\frac{\bar{L}}{C_l - y_l}$, where $\bar{L}$ is the average packet length, and C_l is the actual capacity of link l

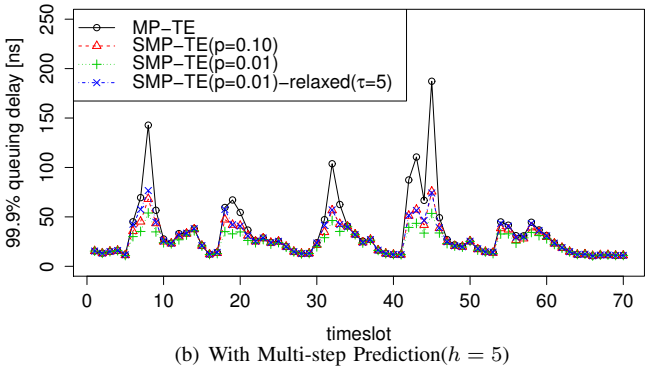
In Fig. 5, SMP-TE achieves lower delay in both cases of one-step prediction and multi-step prediction. This is because SMP-TE sets a safer route that accommodates the traffic without congestion, even when the prediction error occurs. On the other hand, the non-stochastic approaches of simple prediction-based TE and MP-TE cause higher delay, because the routes calculated using only expected traffic no longer deal with unexpected traffic arrival. Of course, congestion may occur with even SMP-TE if the prediction error is significantly outside of the expected range. However, this case only occurs with probability p, which the network operator can set to an appropriate value.

C. Multi-step Prediction Effect

The prediction-based TE can gradually change routes by predicting future congestion in advance. Although this is not an effect of the stochastic approach, our interest is in how this prediction effect is reproduced in our SMP-TE. Figure 6 shows the maximum difference of the path fraction, which is defined as $\max_p |\Delta R_p(t)|$ in each case of the TE method. From this figure, TE with one-step prediction requires a significant route change at time slot 31, but that is avoided by using the multi-step prediction. This is because gradual



(a) With One-step Prediction($h = 1$)



(b) With Multi-step Prediction($h = 5$)

Fig. 5. 99.9% Queuing Delay on the Bottleneck Link at Each Time

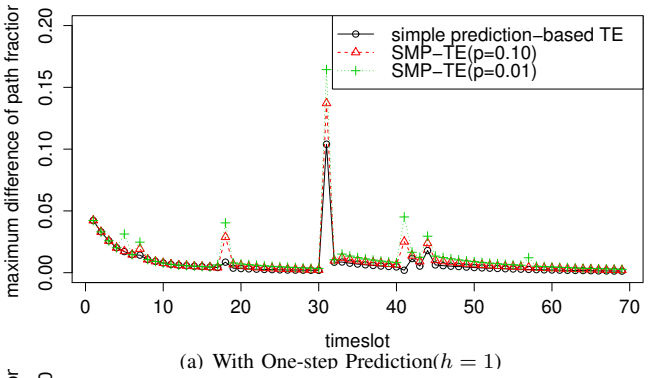
route changes proactively proceeded before the actual traffic change by considering the multi-step prediction. This indicates that far-future prediction is also effective toward avoiding significant route changes in SMP-TE.

However, the frequency of route changes increases in both SMP-TE and MP-TE. This is because wasteful routes changes occur when the predicted future congestion does not actually occur. SMP-TE in particular causes more route changes, because the control server becomes more conservative at expanding future prediction error. One solution of this too-conservative to far-future congestion is relaxation of the future probabilistic constraint, as mentioned in subsection III-B. The effect of constraint relaxation is discussed in the next subsection.

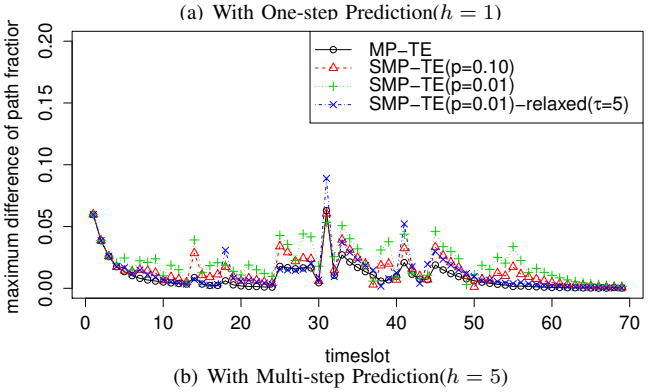
D. Probability Relaxation Effect

In this subsection, we discuss the effect of relaxing future constraints. Figure 6 shows that relaxation avoids frequent route changes, while Fig. 5 shows that congestion is avoided even in the case of the relaxation. To discuss the effect of the probability in more detail, we focus on the route changes performed by the TE methods. Table I shows a summary of routes changes performed by each TE method. In this table, "average" means the average difference of path fraction $|\Delta R_p(t)|$ for all times and all paths, "max" means the maximum difference of the path fractions, and "frequency" means the ratio of time slots in which more than 1% of traffic is moved from a path to other paths. As previously mentioned, the maximum route changes become small in the TE-method using multi-step prediction. On the other hand, more frequent route changes occurred in multi-step prediction, though the average routes changes are the lowest among TE methods.

Comparing the relaxed SMP-TE with original SMP-TE, relaxation of constraints can reduce the frequency of route



(a) With One-step Prediction($h = 1$)



(b) With Multi-step Prediction($h = 5$)

Fig. 6. Maximum Difference of Path Fraction at Each Time

TABLE I
ROUTE CHANGES CAUSED IN EACH TE METHOD

TE using one-step prediction			
	average	max	frequency
simple prediction-based TE	0.083%	10%	16%
SMP-TE($p = 0.1$)	0.094%	14%	23%
SMP-TE($p = 0.01$)	0.11%	16%	36%
TE using multi-step prediction($h = 5$)			
	average	max	frequency
MP-TE	0.074%	6.3%	33%
SMP-TE($p = 0.1$)-relaxed($\tau = 5$)	0.088%	8.9%	42%
SMP-TE($p = 0.1$)-relaxed($\tau = 20$)	0.089%	7.1%	46%
SMP-TE($p = 0.1$)	0.090%	6.0%	51%
SMP-TE($p = 0.01$)-relaxed($\tau = 5$)	0.10%	12%	52%
SMP-TE($p = 0.01$)-relaxed($\tau = 20$)	0.11%	8.9%	62%
SMP-TE($p = 0.01$)	0.012%	6.0%	78%

changes. However, the maximum route changes in relaxed SMP-TE are larger than in the original SMP-TE. This is because constraint relaxation delays the control server reaction to future congestion. With larger τ, the relaxation is gradually conducted over time, avoiding such response delay. However, the result causes frequent route change during ordinary traffic. Therefore, τ should be appropriately tuned to an optimal balance between following traffic variation and ignoring prediction error. Our future work will include developing a method for tuning this parameter.

E. Scalability

Theoretically, the worst-case time complexity of each iteration to solve SOCP is $O(N^2 \sum_i N_i)$ [14] where N is the number of variables and N_i is the dimension of second-order cone constraints. In the SMP-TE, the number of variables is $O(mn^2)$ and the dimension of second-order cone constraints is $O(n^2)$ where m, n and are the number of links and nodes, respectively. Therefore, the computational complexity of SMP-TE is $O(m^2 n^6)$. To use SMP-TE in a large network, we should reduce the calculation time. One approach is decomposing a network into multiple ranges and applying the SMP-TE to each range, which is one of our future research topics.

V. CONCLUSION

We have proposed a TE method called SMP-TE that follows predicted traffic in a way that avoids the impact of prediction error. According to the basic idea of SMPC, our SMP-TE calculates routes while considering the probability distribution of prediction error. Through simulation using actual backbone network traffic traces, we demonstrated that SMP-TE can avoid congestion in cases where simple prediction-based TE cannot. Additional route changes required to accommodate the prediction error remained small.

Future work will include further verification of our method using larger networks with more realistic traffic. Furthermore, we will reduce calculation times by adapting SMP-TE to distributed control that determines routes using only local traffic information. We are now implementing SMP-TE, and will report the results in a forthcoming paper.

ACKNOWLEDGEMENT

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

REFERENCES

[1] M. Roughan, "Robust network planning," *Guide to Reliable Internet Services and Applications Conputer Communications and Networks*, pp. 137–177, 2010.
[2] S. Kandula, D. Katabi, B. Davie, and A. Charny, "Walking the tightrope: responsive yet stable traffic engineering," in *Proceedings of ACM SIGCOMM 2005*, Aug. 2005, pp. 253–264.
[3] H. Wang, H. Xie, L. Qiu, Y. R. Yang, Y. Zhang, and A. Greenberg, "COPE: traffic engineering in dynamic networks," in *Proceedings of ACM SIGCOMM 2006*, vol. 36, no. 4, Aug. 2006, pp. 99–110.
[4] N. Wang, K. H. Ho, G. Pavlou, and M. Howarth, "An overview of routing optimization for Internet traffic engineering," *IEEE Communications Survey & Tutorials*, vol. 10, no. 1, pp. 36–56, first quarter 2008.
[5] K. Srijith, L. Jacob, and A. Ananda, "TCP Vegas-A: Improving the performance of TCP Vegas," *Computer Communications*, vol. 28, no. 4, pp. 429–440, Mar. 2005.
[6] Tatsuya Otoshi, "Prediction-based control theoretic approach for robust traffic engineering," Master's thesis, Graduate School of Information Science and Technology, Osaka University, February 2014.
[7] T. Hashimoto, "Probabilistic constrained model predictive control for linear discrete-time systems with additive stochastic disturbances," in *Proceedings of the 52nd IEEE Conference on Decision and Control*, Dec. 2013, pp. 6434–6439.
[8] "Internet2 data," available from http://internet2.edu/observatory/archive/data-collections.html.
[9] M. F. Zhani, H. Elbiaze, and F. Kamoun, "Analysis and prediction of real network traffic," *Journal of Networks*, vol. 4, no. 9, pp. 855–865, Nov. 2009.
[10] K. Papagiannaki, N. Taft, Z.-L. Zhang, and C. Diot, "Long-term forecasting of Internet backbone traffic," *IEEE Transactions on Neural Network*, vol. 16, no. 5, pp. 1110–1124, Sep. 2005.
[11] S. Han-Lin, J. Yue-Hui, C. Yi-Dong, and C. Shi-Duan, "Network traffic prediction by a wavelet-based combined model," *Chinese Physics B*, vol. 18, no. 11, pp. 4760–4768, Nov. 2009.
[12] M. Johnston, H.-W. Lee, and E. Modiano, "Robust network design for stochastic traffic demands," *Journal of Lightwave Technology*, vol. 31, no. 18, pp. 3104–3116, Sep. 2013.
[13] "IBM ILOG CPLEX Optimizer," optimization software : http://www-01.ibm.com/software/integration/optimization/cplex-optimizer.
[14] M. S. Lobo, L. Vendenbeghe, S. Boyd, and H. Lebret, "Applications of second-order cone programming," *Linear Algebra and its Applications*, vol. 284, no. 1, pp. 193–228, Nov. 1998.