Time-Aware Congestion-Free Routing Reconfiguration

Shih-Hao Tseng, Chiun Lin Lim, Ning Wu, and Kevin Tang

School of Electrical and Computer Engineering, Cornell University

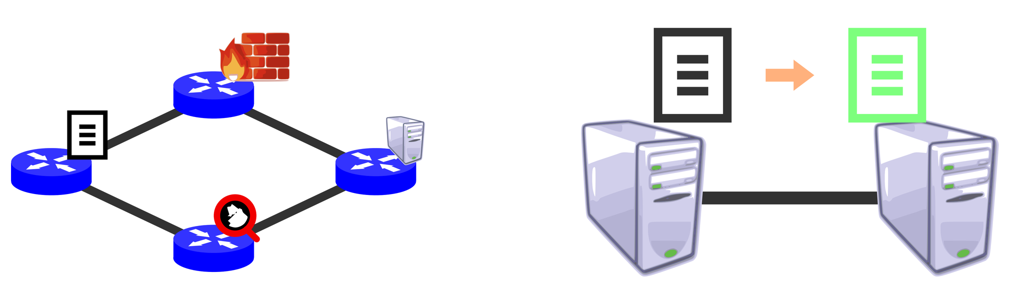
Supported by NSF grant CNS-1544761

Objective and Key Contributions

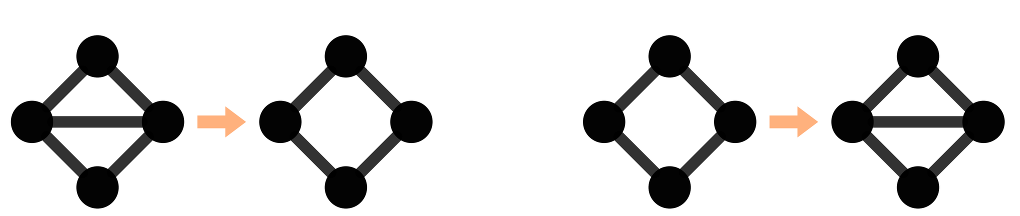
- Instead of finding a least step update, timing information is incorporated into the new model to pursue a fastest update within given number of steps.
- Finding the fastest update is shown NP-hard.
- Efficient algorithm is developed to achieve fast update with performance guarantee.

Introduction

Network operators change the network configuration to respond to the anticipated requests.



(a) middlebox traversal constraint satisfaction (b) virtual machine live migration



(c) scheduled network maintenance (d) load balancing after network upgrade

Figure 1: Anticipated network changes.

A sequence of update steps prevents the potential congestion happening during an one-shot update [1]. The state-of-the-art solution ignores timing information and solves for the least step solution. However, a least step solution does not necessarily translate to a least time update. Our goal is to find the fastest congestion-free routing reconfiguration in a given number of steps.

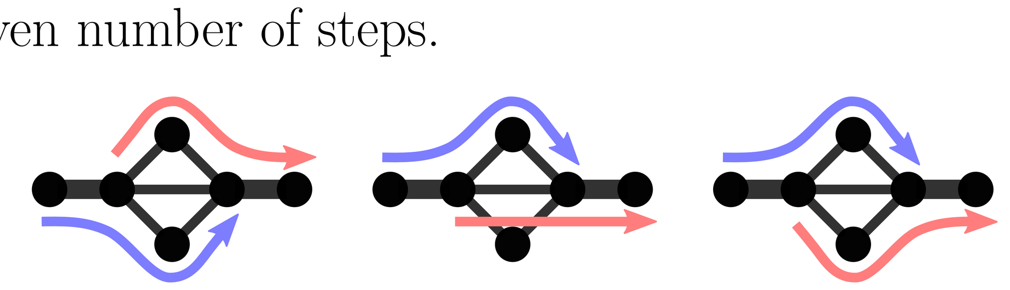
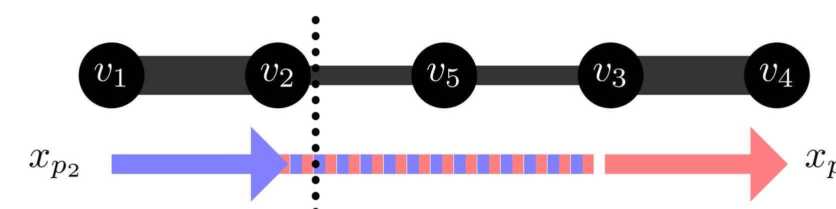


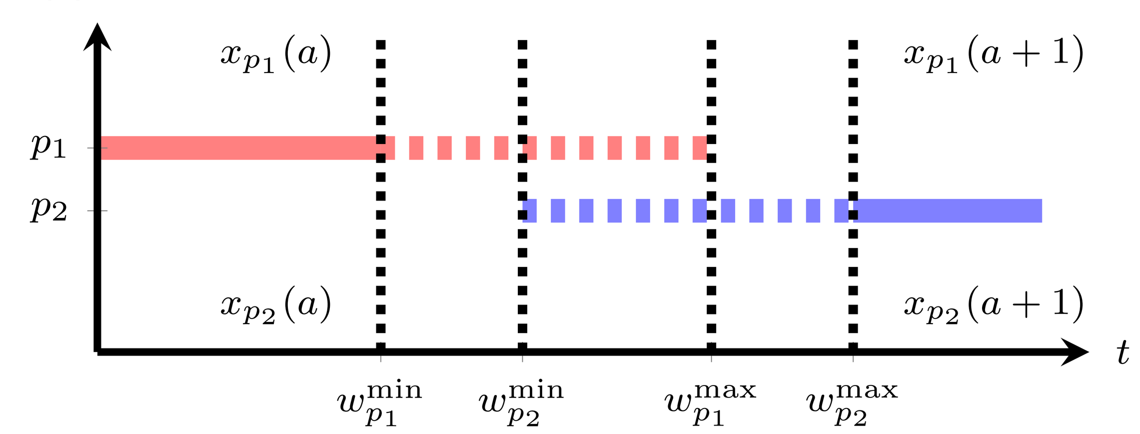
Figure 2: Multi-step update prevents congestion.

Model

We incorporate timing information into the model as intervals.



(a) The traffic rate x_{p_1} and x_{p_2} change between step a and step $a+1$



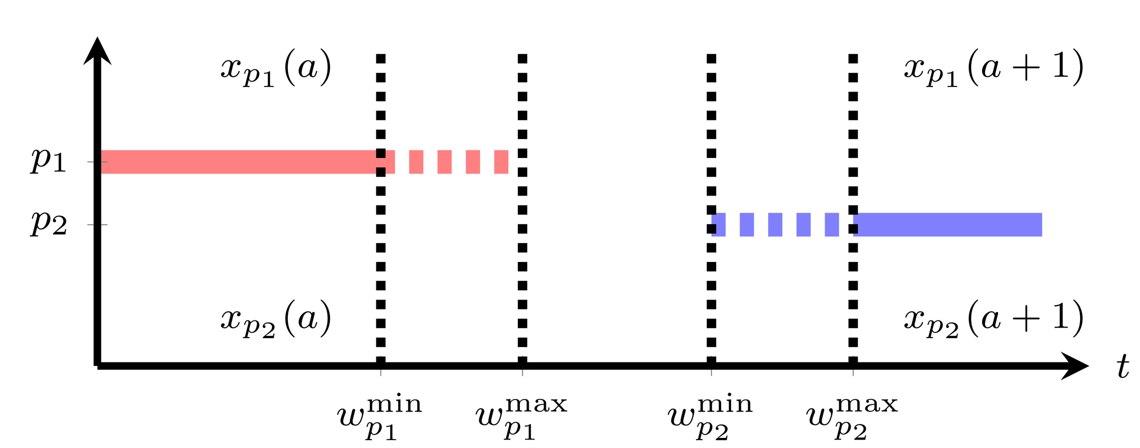
(b) Intervals are introduced to model the uncertainty, which can overlap with one another

Figure 3: Uncertainty intervals reflect the accuracy of the timing information given to the operator

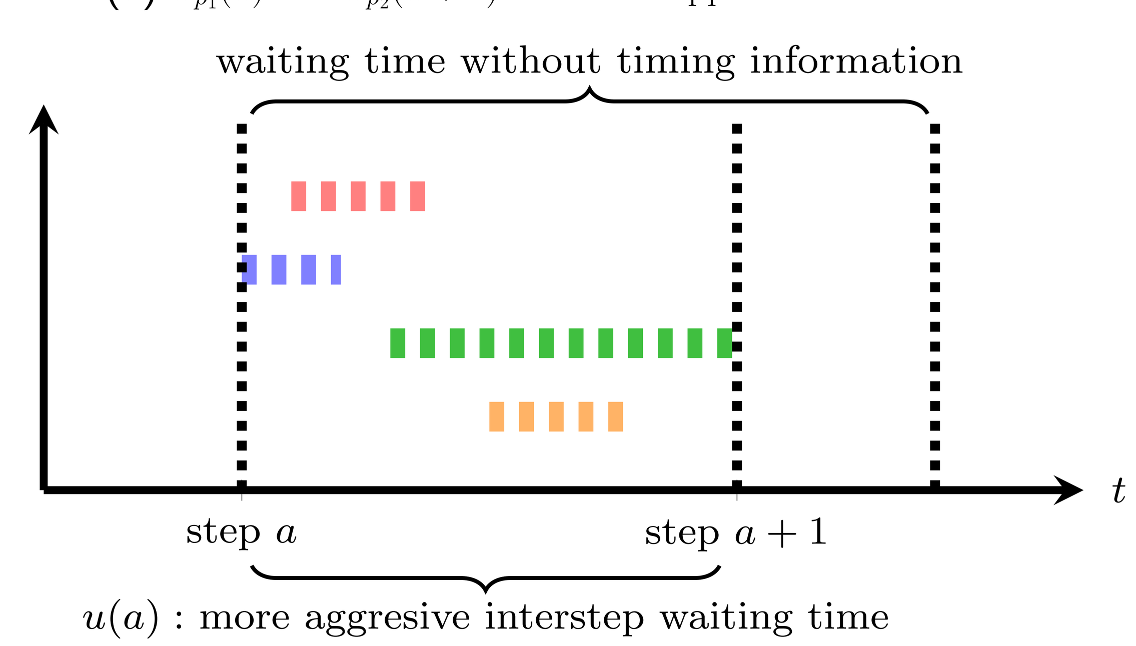
Timing information helps us

1. rule out the impossible cases
2. find the shortest waiting time between steps

and hence we can update more aggressively.



(a) $x_{p_1}(a)$ and $x_{p_2}(a+1)$ will never appear at the same time



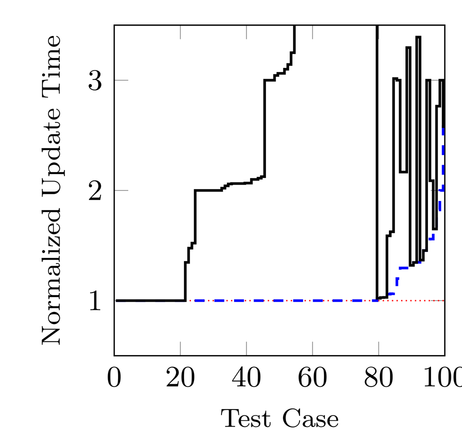
(b) $u(a)$ is shorter than the time-oblivious waiting time

Figure 4: Timing information leads to faster update

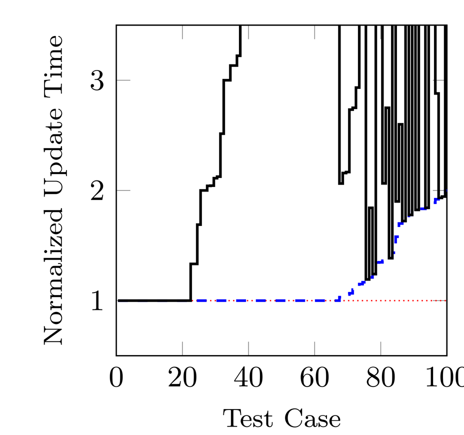
Simulation Results

We compare our method with SWAN [1] and zUpdate [2] under different network topologies.

(a) B4 topology

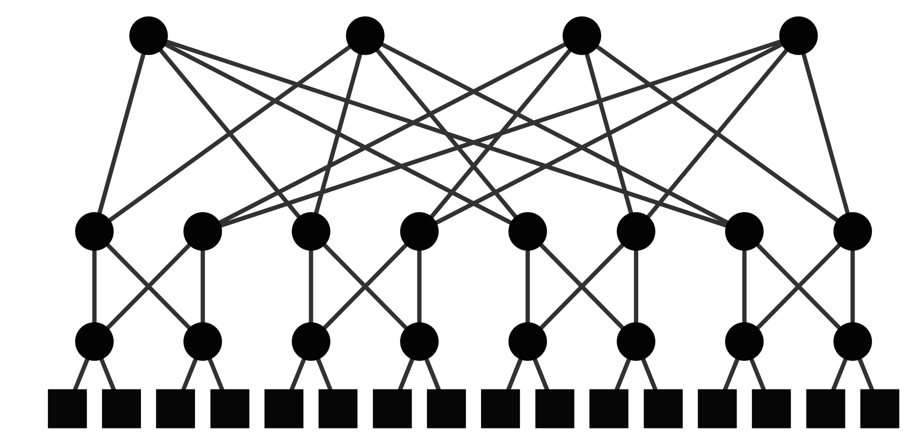


(b) $\lambda = 0.1\%$

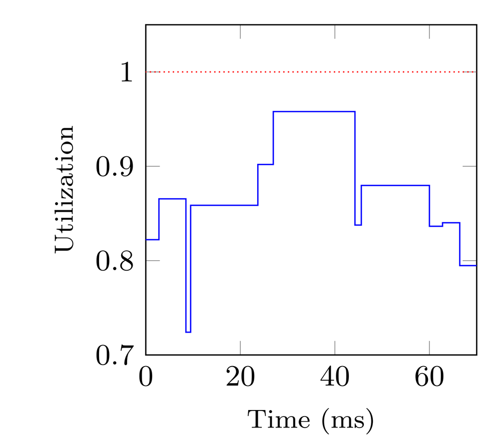


(c) $\lambda = 0.05\%$

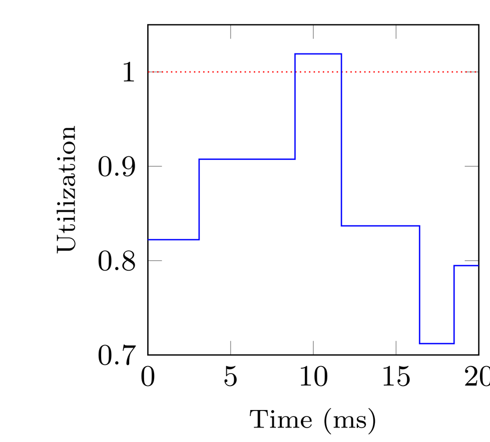
Figure 5: Comparison with SWAN [1], where λ is the scratch capacity rate. Our method not only outperforms SWAN but also achieves near-optimal performance. (blue dashed: our method; black solid: SWAN; red dotted: optimal)



(a) Fat-tree topology



(b) Our method



(c) zUpdate

Figure 6: Comparison with zUpdate [2]. Our method ensures the update is congestion-free, while zUpdate may exceed the link capacity under the presence of uncertainty. (the utilization is normalized by the link capacity)

Optimization Problem

The Fast Congestion-free Reconfiguration (FCR) problem can be expressed in the following form

$$
\begin{aligned}
\mathrm{FCR}(b) = \quad & \text{minimize} \quad \sum_{a=0}^{b-1} u(a) \\
& \text{subject to} \quad \text{link capacity constraints} \\
& \qquad\qquad\quad \text{demand constraints} \\
& \qquad\qquad\quad \text{boundary constraints} \\
& \qquad\qquad\quad \text{integer constraints for } u
\end{aligned}
$$

where the integer constraints are the key reason why the problem is NP-hard. We propose an approximation algorithm by transforming the problem and relaxing the integer constraints, which has proven approximation ratio and is shown effective via extensive simulations.

Conclusion

- The time-aware optimization model finds fast congestion-free routing reconfiguration plans.
- The proposed efficient approximation algorithm approaches the NP-hard problem with performance guarantee.
- Extensive packet-level simulations demonstrate the effectiveness of the algorithms.

References

[1] Chi-Yao Hong, Srikanth Kandula, et al. Achieving high utilization with software-driven WAN. *ACM SIGCOMM CCR*, 43(4):15–26, 2013.

[2] Hongqiang Harry Liu, Xin Wu, et al. zUpdate: Updating data center networks with zero loss. *ACM SIGCOMM CCR*, 43(4):411–422, 2013.