

Time-Aware Congestion-Free Routing Reconfiguration

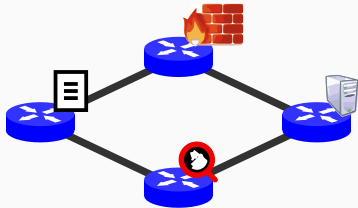
Shih-Hao Tseng, (pronounced as “She-How Zen”)
joint work with Chiun Lin Lim, Ning Wu, and Kevin Tang

May 17, 2016

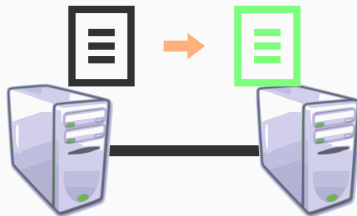
School of Electrical and Computer Engineering, Cornell University

Introduction

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



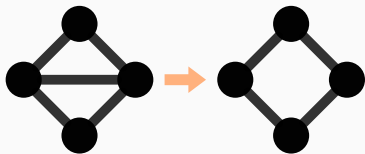
(a) middlebox traversal constraint satisfaction



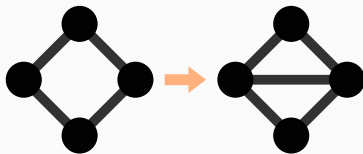
(b) virtual machine live migration

Introduction

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

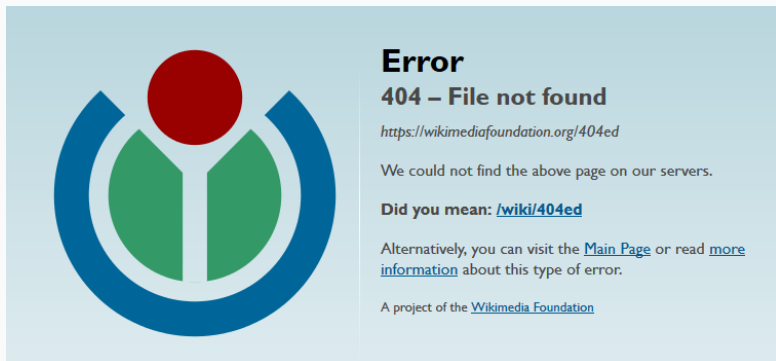


(c) scheduled network maintenance



(d) load balancing after network upgrade

- The reconfiguration should be done seamlessly.



Source: <https://wikimediafoundation.org/404ed>

- The reconfiguration should be done seamlessly.



Summary of the Amazon EC2 and Amazon RDS Service Disruption in the US East Region

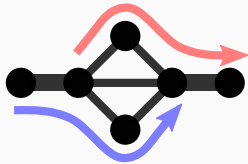
April 29, 2011

Now that we have fully restored functionality to all affected services, we would like to share more details with our customers about the events that occurred with the Amazon Elastic Compute Cloud ("EC2") last week, our efforts to restore the services, and what we are doing to prevent this sort of issue from happening again. We are very aware that many

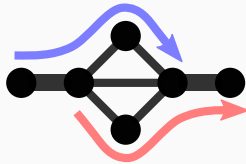
Source: <http://aws.amazon.com/message/65648>

Introduction

- A sequence of update steps prevents the potential congestion happening during an one-shot update (Hong et al., 2013).
- The state-of-the-art solution ignores timing information and solves for the least step solution.



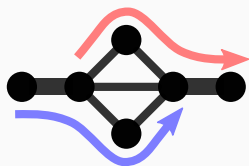
step 0



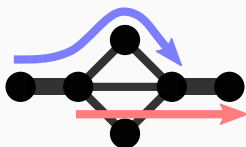
step b

Introduction

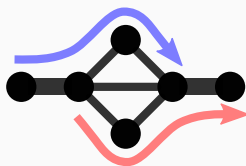
- A sequence of update steps prevents the potential congestion happening during an one-shot update (Hong et al., 2013).
- The state-of-the-art solution ignores timing information and solves for the least step solution.



step 0



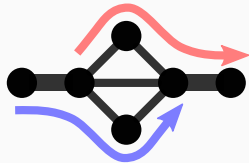
step 1 & step 2



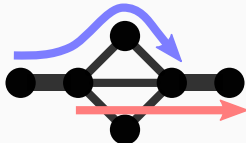
step b

Introduction

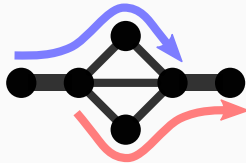
- However, a least step solution does not necessarily translate to a least time update.
- Our goal is to find the fastest congestion-free routing reconfiguration in a given number of steps.



step 0



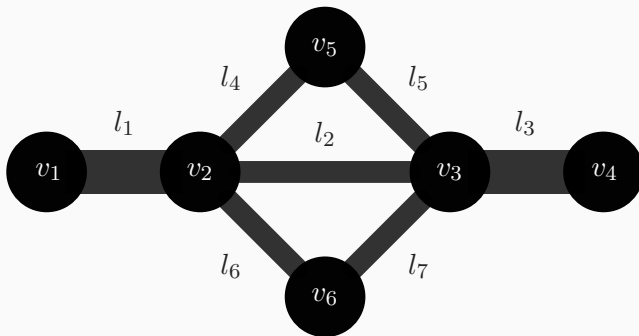
step 1 & step 2



step b

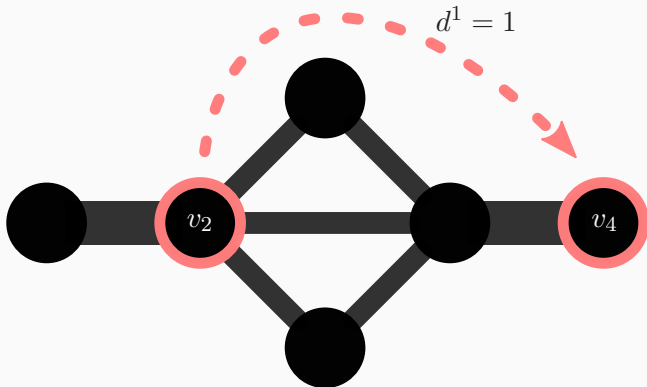
Model

- The set of switches $V = \{v_1, v_2, \dots, v_6\}$.
- The set of directed links $L = \{l_1, l_2, \dots, l_7\}$.



Model

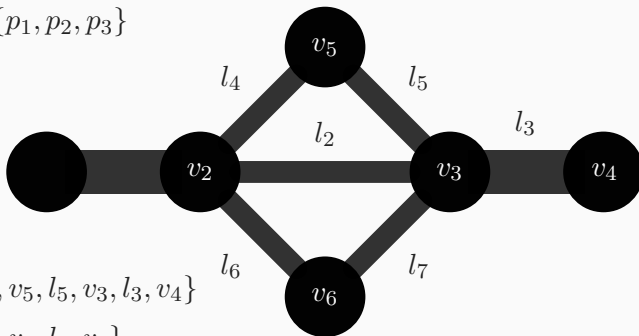
- Each user $n \in N$ demands a traffic rate d^n from its source to its destination.



Model

- A set of acyclic paths P^n is predetermined and established for each user n .

$$P^1 = \{p_1, p_2, p_3\}$$



$$p_1 = \{v_2, l_4, v_5, l_5, v_3, l_3, v_4\}$$

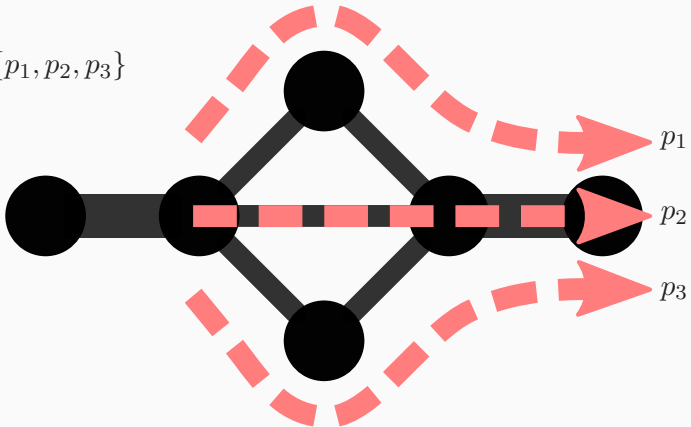
$$p_2 = \{v_2, l_2, v_3, l_3, v_4\}$$

$$p_3 = \{v_2, l_6, v_6, l_7, v_3, l_3, v_4\}$$

Model

- A set of acyclic paths P^n is predetermined and established for each user n .

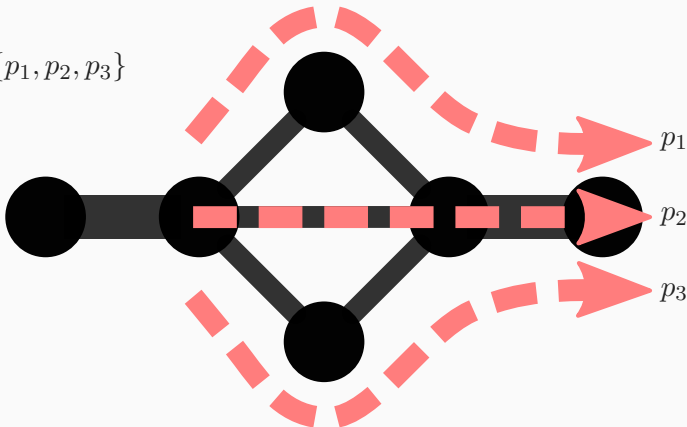
$$P^1 = \{p_1, p_2, p_3\}$$



Model

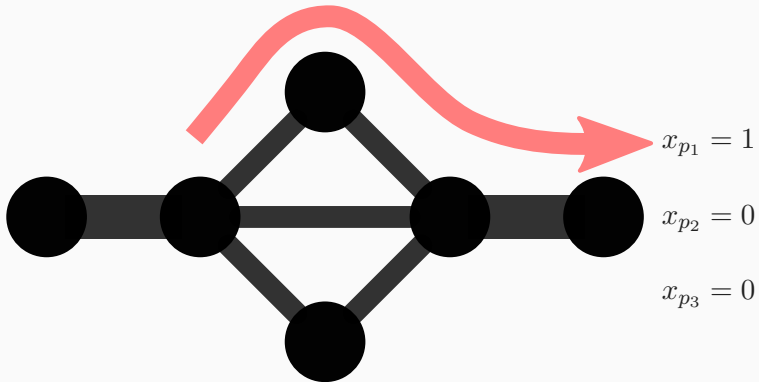
- The controller performs routing by specifying the split ratio among the paths in P^n for each user n at its source.

$$P^1 = \{p_1, p_2, p_3\}$$



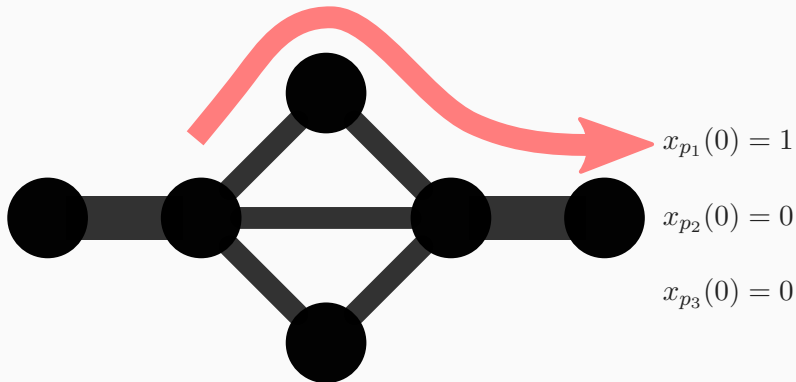
Model

- x_p represents the traffic rate of the flow along a path p .



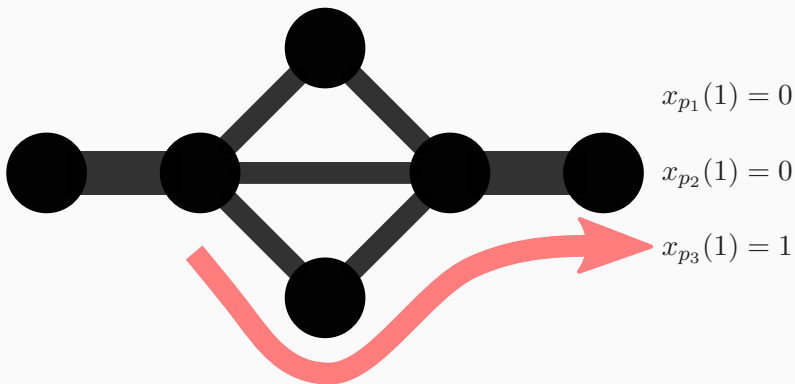
Model

- The parenthesized step number a is attached after a variable to refer to its value between step a and $a + 1$.



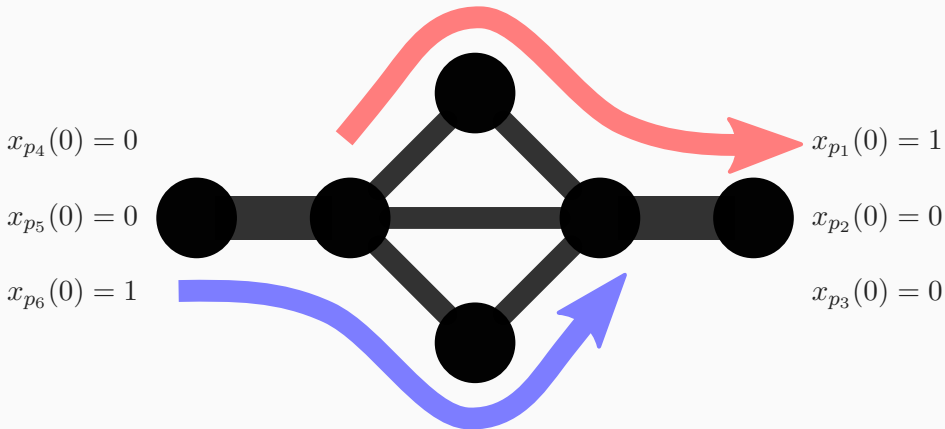
Model

- The parenthesized step number a is attached after a variable to refer to its value between step a and $a + 1$.



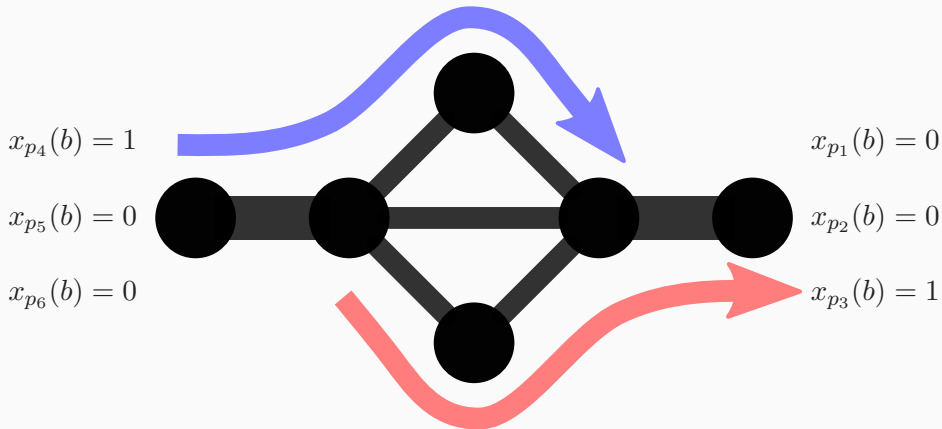
Model

- The initial and target configurations are specified by $x_p(0)$ and $x_p(b)$ for all paths $p \in P = \bigcup_{n \in N} P^n$.



Model

- The initial and target configurations are specified by $x_p(0)$ and $x_p(b)$ for all paths $p \in P = \bigcup_{n \in N} P^n$.



Fast Congestion-free Reconfiguration

FCR(b) = minimize

$$\text{subject to } \sum_{p \in P^n} x_p(a) = d^n \quad \forall n \in N, 1 \leq a \leq b-1$$

$$x_p(a) \geq 0 \quad \forall p \in P, 1 \leq a \leq b-1$$

$$x_p(0), x_p(b) \text{ are given } \quad \forall p \in P$$

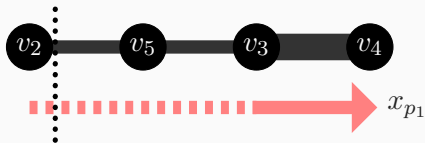
Fast Congestion-free Reconfiguration

FCR(b) = minimize

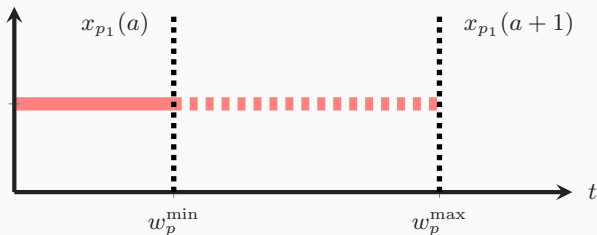
subject to demand constraints

boundary constraints

- We incorporate timing information into the model as intervals.

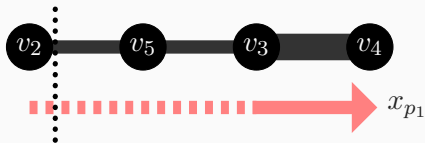


(a) The traffic rate x_{p1} changes between step a and step $a + 1$

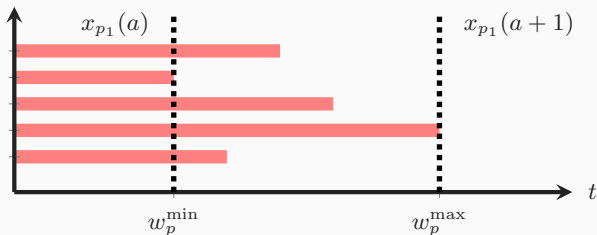


(b) An interval is introduced to model the uncertainty

- We incorporate timing information into the model as intervals.

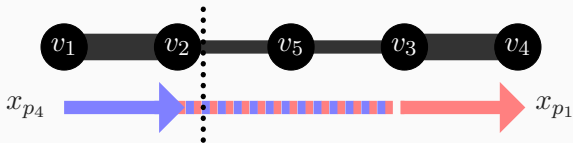


(a) The traffic rate x_{p_1} changes between step a and step $a + 1$

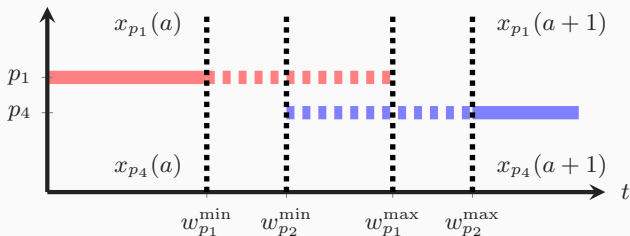


(c) The timing of the change is uncertain

- We incorporate timing information into the model as intervals.



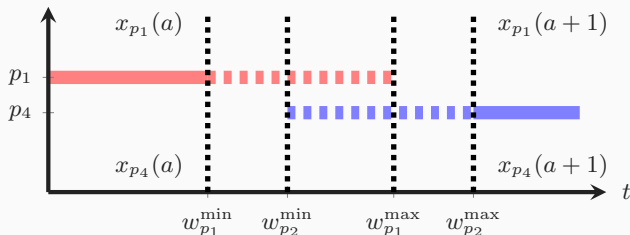
(d) Two paths p_1 and p_4 share some common links



(e) Uncertainty interval overlapping determines possible mixes of the flows

- We incorporate timing information into the model as intervals.

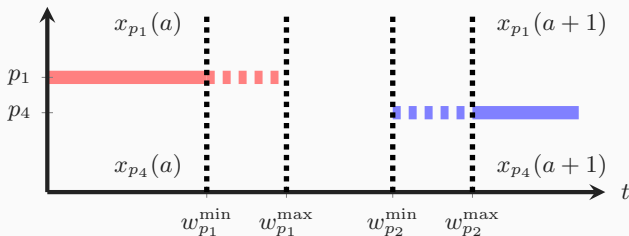
$$\begin{aligned}x_{p_1}(a) + x_{p_4}(a) &\leq 1 & x_{p_1}(a+1) + x_{p_4}(a) &\leq 1 \\x_{p_1}(a) + x_{p_4}(a+1) &\leq 1 & x_{p_1}(a+1) + x_{p_4}(a+1) &\leq 1\end{aligned}$$



(e) Uncertainty interval overlapping determines possible mixes of the flows

- We incorporate timing information into the model as intervals.

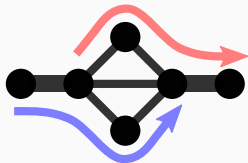
$$\begin{aligned}x_{p_1}(a) + x_{p_4}(a) &\leq 1 & x_{p_1}(a+1) + x_{p_4}(a) &\leq 1 \\x_{p_1}(a+1) + x_{p_4}(a+1) &\leq 1\end{aligned}$$



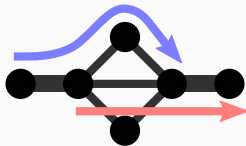
(e) Uncertainty interval overlapping determines possible mixes of the flows

Model

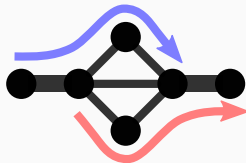
- Timing information helps us rule out the impossible cases and hence we can update more aggressively.



step 0



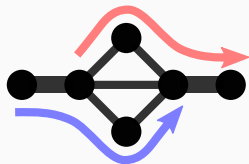
step 1 & step 2



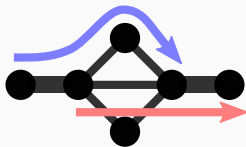
step b

Model

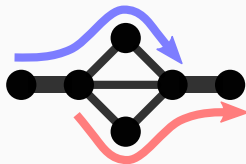
- Timing information helps us rule out the impossible cases and hence we can update more aggressively.



step 0



step 1



step b

Fast Congestion-free Reconfiguration

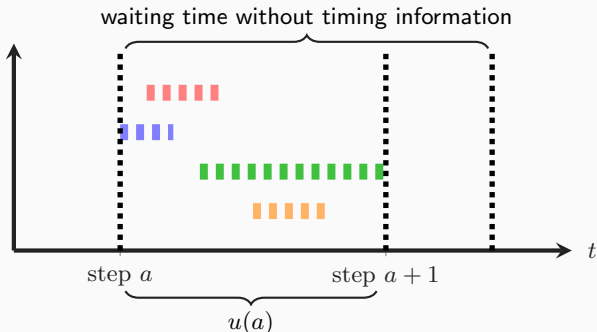
FCR(b) = minimize

subject to link capacity constraints

demand constraints

boundary constraints

- Timing information helps us find the shortest waiting time between steps and hence we can update more aggressively.



$$\begin{aligned}u(a) &= \max \{w_p^{\max} : \text{flow along } p \text{ changes between step } a \text{ and } a+1\} \\ &= \max_p \{w_p^{\max} z_p(a)\}\end{aligned}$$

Fast Congestion-free Reconfiguration

$$\begin{aligned} \text{FCR}(b) = & \text{minimize} && \sum_{a=0}^{b-1} u(a) \\ & \text{subject to} && \text{link capacity constraints} \\ & && \text{demand constraints} \\ & && \text{boundary constraints} \\ & && u(a) \geq w_p^{\max} \cdot z_p(a) \quad \forall p \in P, 0 \leq a \leq b-1 \\ & && z_p(a) \cdot \alpha_p \geq |x_p(a+1) - x_p(a)| \\ & && \forall p \in P, 0 \leq a \leq b-1 \\ & && z_p(a) \in \{0, 1\} \quad \forall p \in P, 0 \leq a \leq b-1 \end{aligned}$$

Fast Congestion-free Reconfiguration

$$\text{FCR}(b) = \text{minimize } \sum_{a=0}^{b-1} u(a)$$

subject to link capacity constraints
demand constraints
boundary constraints
integer constraints for u

Fast Congestion-free Reconfiguration

$$\text{FCR}(b) = \text{minimize } \sum_{a=0}^{b-1} u(a)$$

subject to link capacity constraints

demand constraints

boundary constraints

integer constraints for u

NP-hardness

Fast Congestion-free Reconfiguration

$$\text{FCR}(b) = \text{minimize } \sum_{a=0}^{b-1} u(a)$$

subject to link capacity constraints
demand constraints
boundary constraints
relaxed constraints for u

- We compare our method with other methods (SWAN and zUpdate) under different network topologies.
- The step upper bound b is set to 10.
- We randomly select pairs of nodes as users sending traffic flow with uniformly distributed size. Each user utilizes 2 acyclic paths predetermined by Yen's k -shortest-path Algorithm with $k = 2$ to send traffic through.

Inter-Datcenter Network: B4



Inter-Datcenter Network: B4



Inter-Datcenter Network: B4



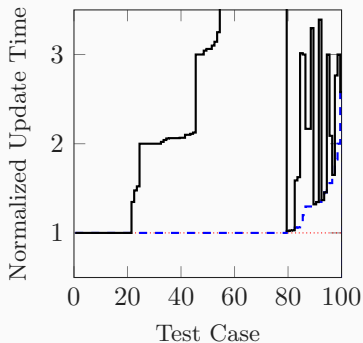
Inter-Datcenter Network: B4



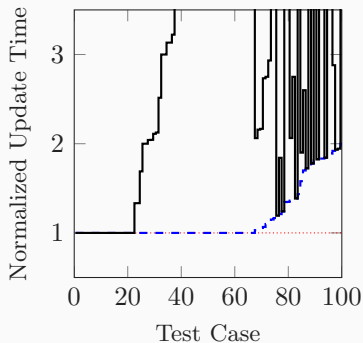
Inter-Datcenter Network: B4



Inter-Datcenter Network: B4



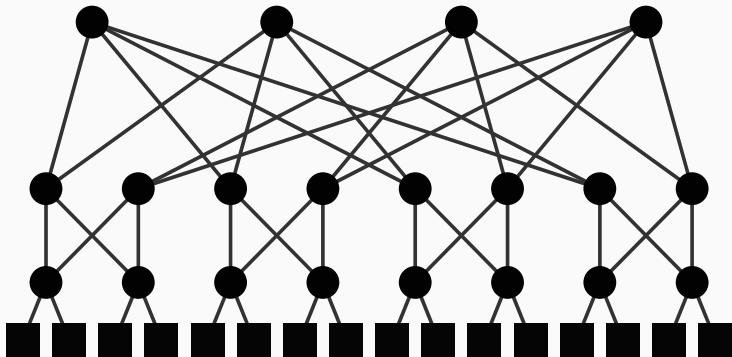
(a) Scratch capacity rate $\lambda = 10\%$



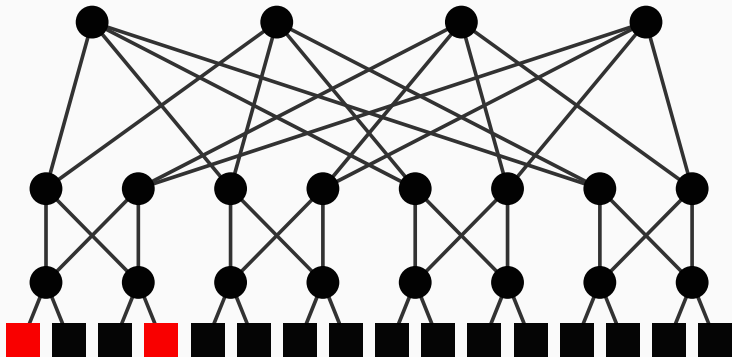
(b) Scratch capacity rate $\lambda = 5\%$

Figure 1: The resulted update time normalized by the shortest update time. Dashed line: our method; normal line: SWAN; dotted constant 1 line: optimal.

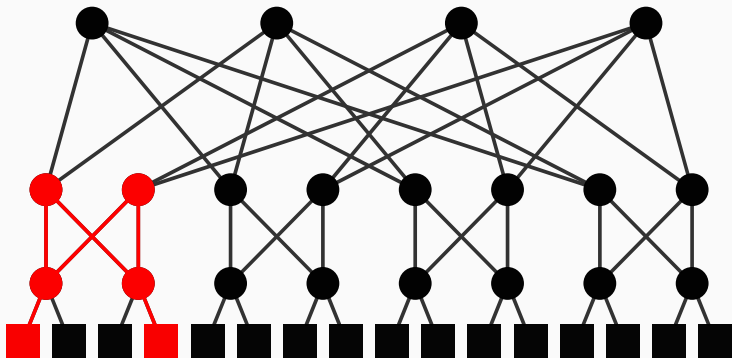
Intra-Datcenter Network: Fat-Tree



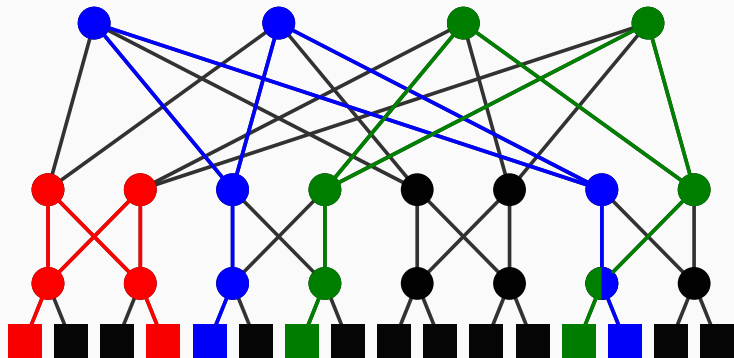
Intra-Datcenter Network: Fat-Tree



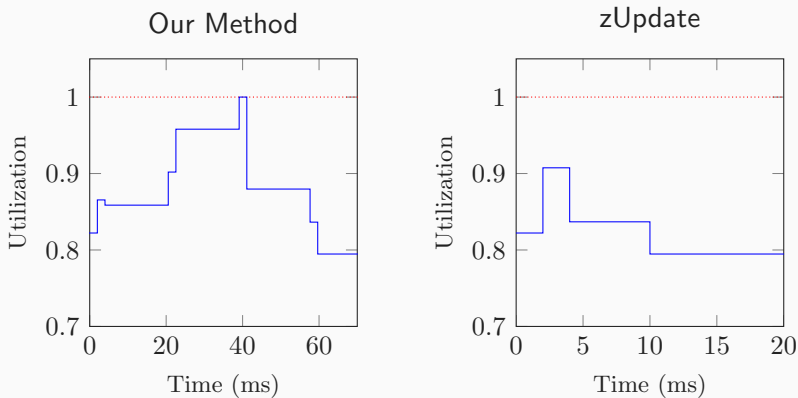
Intra-Datcenter Network: Fat-Tree



Intra-Datcenter Network: Fat-Tree

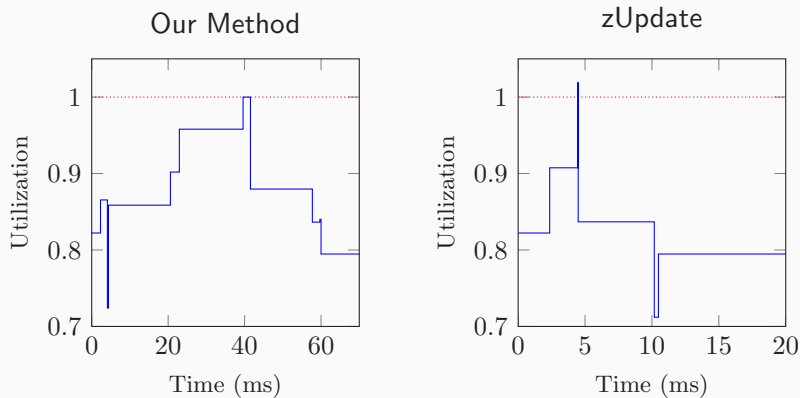


Intra-Datcenter Network: Fat-Tree



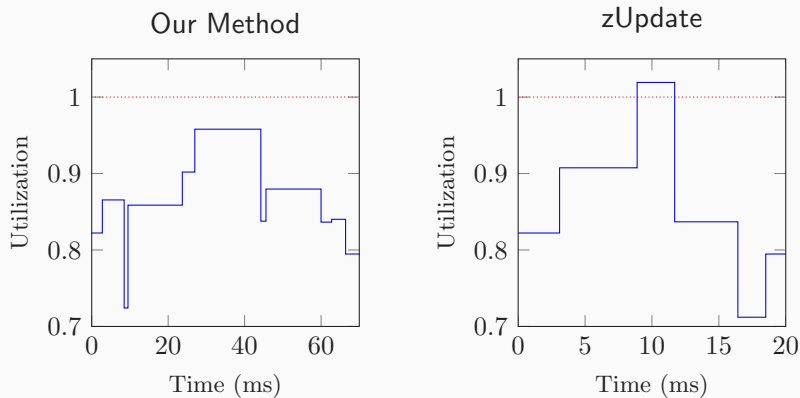
(a) Without uncertainty: Both our method and zUpdate are congestion-free

Intra-Datcenter Network: Fat-Tree



(b) Low uncertainty: Our method is congestion-free while zUpdate congests

Intra-Datcenter Network: Fat-Tree



(c) High uncertainty: Our method remains congestion-free and zUpdate endures a long congestion period

Comparison with State-of-the-art Methods

Method	Our Method	Hong et al., 2013 (SWAN)	Liu et al., 2013 (zUpdate)	Jin et al., 2014 (Dionysus)
Applicable Network	arbitrary	arbitrary	layered structure	arbitrary
Update Objective	minimum time	minimum step	minimum step	maximum parallelism
Applicable Routing	tunnel-based	tunnel-based	switch-based	tunnel-based switch-based
Uncertainty Tolerance	yes	yes	no	yes
Dependency Knowledge	unneeded	unneeded	unneeded	required

Comparison with State-of-the-art Methods




- SWAN considers only the order-oblivious case when the network is totally uncertain for the operator, which is just an extreme case of our framework, happening when all the uncertainty intervals overlap with one another.
- zUpdate assumes the network has a layered structure with known propagation delays between the layers and no switch processing delay is considered. Hence, it is another special case of our framework with uncertainty issue eliminated.

Conclusion

- We formulate a time-aware optimization model to find fast congestion-free routing reconfiguration plans.
- This framework helps determine less conservative update schedule.
- We further provide an efficient approximation algorithm to solve this new optimization problem, which is proven to be NP-hard, with performance guarantee.
- Extensive packet-level simulations confirm our predictions.

Questions & Answers

References

-  C.-Y. Hong, S. Kandula, R. Mahajan, M. Zhang, V. Gill, M. Nanduri, and R. Wattenhofer, “Achieving high utilization with software-driven WAN,” *ACM SIGCOMM CCR*, vol. 43, no. 4, pp. 15–26, 2013.
-  H. H. Liu, X. Wu, M. Zhang, L. Yuan, R. Wattenhofer, and D. Maltz, “zUpdate: Updating data center networks with zero loss,” *ACM SIGCOMM CCR*, vol. 43, no. 4, pp. 411–422, 2013.
-  X. Jin, H. H. Liu, R. Gandhi, S. Kandula, R. Mahajan, M. Zhang, J. Rexford, and R. Wattenhofer, “Dynamic scheduling of network updates,” in *Proc. ACM SIGCOMM*, 2014, pp. 539–550.