Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

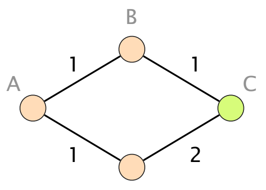
Networked Systems
ETH Zürich — seit 2015

Communication Networks

Prof. Laurent Vanbever

Exercises week 7 – Internet Routing

Convergence (Exam Style Question)



Loopy or not?

Consider this simple network running OSPF as link-state routing protocol. Each link is associated with a weight that represents the cost of using it to forward packets. Link weights are bi-directional.

Assume that routers A, B and D transit traffic for an IP destination connected to C and that link (B, C) fails. Which nodes among A, B and D could potentially see their packets being stuck in a transient forwarding loop? Which ones would not?

Assume now that the network administrator wants to take down the link (B, C), *on purpose*, for maintenance reasons. To avoid transient issues, the administrator would like to move away all traffic from the link *before* taking it down and this, without creating any transient loop (if possible). What is the minimum sequence of increased weights setting on link (B, C) that would ensure that *no packet* destined to C is dropped?

Exam Question

For the following statements, decide if they are *true* or *false*. Motivate your decision. These questions are directly taken from the Communication Networks final exam of 2016.

a) Consider a positively weighted graph G. Applying the Bellman-Ford (used by distance-vector protocols) or Dijkstra (used by link-state protocols) algorithm on G would lead to the same forwarding state.

b) Link-state protocols (such as OSPF) are guaranteed to compute loop-free forwarding state as long as the link-state databases are consistent on all routers.

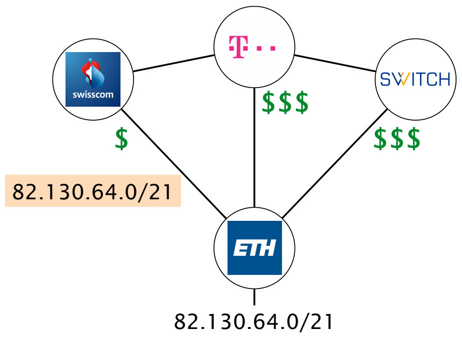
c) Link-state protocols (such as OSPF) require routers to maintain less state than distance-vector protocols (such as RIP).

d) Poisoned reverse solves the problem of count-to-infinity.

e) Consider a positively weighted graph G. Multiplying all link weights by 2 would change the all-pairs shortest paths computed by the Dijkstra algorithm on G.

f) Consider a positively weighted graph G. Adding 1 to all link weights would change the all-pairs shortest paths computed by the Dijkstra algorithm on G.

Traffic Engineering



ETH is connected to three providers with different costs.

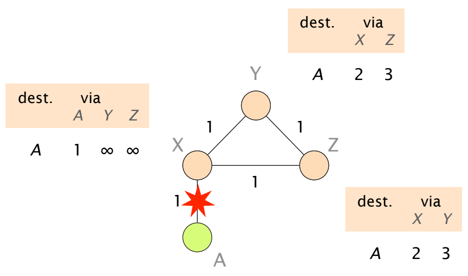
Assume that ETH has only one prefix: 82.130.64.0/21. As depicted on the left, the ETH network is connected to three providers (Swisscom, Deutsche Telekom and Switch) and the providers are interconnected with each other. The contract with Swisscom is the cheapest one (indicated by the dollar symbols). For this reason, ETH wants to receive all the incoming traffic over the Swisscom link and therefore announces its prefix only to Swisscom.

a) Do you think that is a good configuration? What happens if the link between ETH and Swisscom fails?

b) To improve the connectivity in case of a link failure between ETH and Swisscom, ETH wants to optimize its announcements. Write down the prefixes which ETH announces to Swisscom, Deutsche Telekom and Switch. During normal operation (no link failure) ETH should still receive all incoming traffic over the Swisscom link.

c) After further investigations, ETH decides that only traffic towards 82.130.68.0/23 has to be received over the Swisscom link. All the other traffic can enter over any of the providers. Which prefixes do you have to announce to achieve this traffic distribution?

Convergence with Poisoned Reverse



Consider the network on the left which uses distance vector routing with poisoned reverse. Each link is associated with a weight that represents the cost of using it to forward packets. Link weights are bi-directional.

Assume that the link between X and A fails (as shown in the figure) and use the table below to show the first 8 steps of the convergence process. How many steps does it take until the network has converged to a new forwarding state? Explain your observations.

		X			**Y**		**Z**	
	dst=A	via A	via Y	via Z	via X	via Z	via X	via Y
$t = 0$	before the failure	1	∞	∞	2	3	2	3
$t = 1$	after X sends its vector	★						
$t = 2$	after Y sends its vector							
$t = 3$	after Z sends its vector							
$t = 4$	after X sends its vector							
$t = 5$	after Y sends its vector							
$t = 6$	after Z sends its vector							
$t = 7$	after X sends its vector							
$t = 8$	after Y sends its vector							

Add the distance vectors to this table