ETH zürich

Networked Systems
ETH Zürich — seit 2015

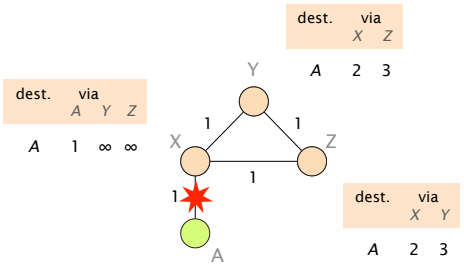
Communication Networks

Prof. Laurent Vanbever

Exercise 6 – Convergence Process in Link-State and Distance-Vector Routing

Convergence

6.1 Convergence with Poisoned Reverse



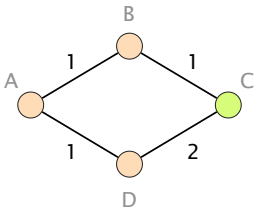
Consider the network on the left which uses distance vector routing with poisoned reverse. Each link is associated with a weight that represents the cost of using it to forward packets. Link weights are bi-directional.

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

6.2 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?