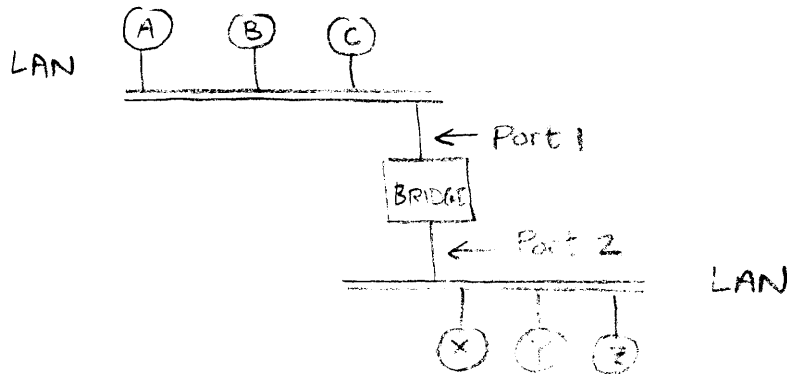


EXTENDING LANs

- BRIDGES ARE USED TO EXTEND LANs

- C.g. IN ETHERNET WE HAVE PHYSICAL LIMITATIONS ON CABLE LENGTH



- BRIDGE FORWARDS PACKETS ON ALL PORTS, EXCEPT ONE IF HEARD PACKET ON
- BUT A BRIDGE DOES NOT HAVE TO FORWARD ALL PACKETS (e.g. A to B)

CONFIGURING A BRIDGE

- IF A SENDS A PACKET TO B, BRIDGE DOES NOT NEED TO FORWARD ON PORT 2.

- BRIDGE NEEDS TO KNOW THAT B IS ON PORT 1

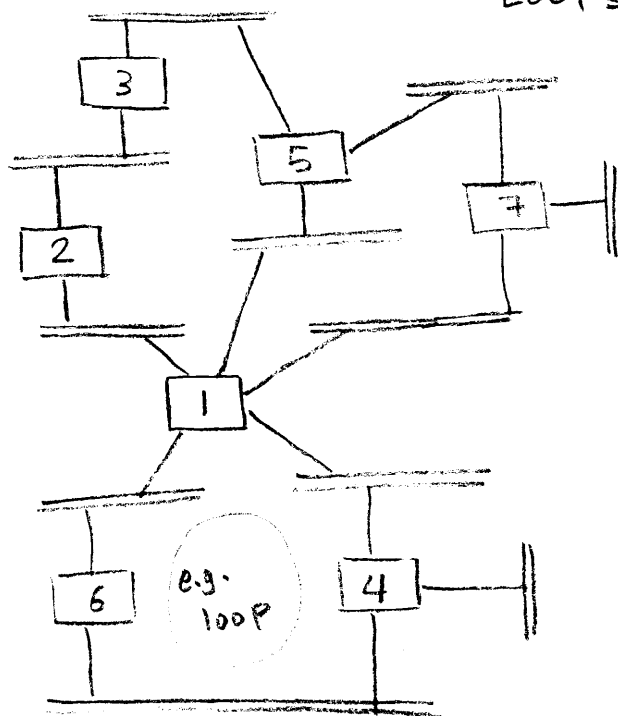
- BRIDGE CAN BE CONFIGURED WITH A TABLE

HOST	PORT
A	1
B	1
C	1
X	2
Y	2
Z	2

LEARNING THE TABLE

- A BRIDGE CAN LEARN THE TABLE ON ITS OWN | HOW?
- THE SOURCE ADDRESS IS USUALLY IN THE PACKET HEADER, C.G. ETHERNET
- WHEN BRIDGE SEES A PACKET FROM A GIVEN SOURCE ON A GIVEN PORT, IT RECORDS THE SOURCE/PORT INFO IN THE TABLE
 - TIME OUT IS ASSOCIATED WITH EACH ENTRY, ENTRY IS DISCARDED AFTER THAT, C.G. HOST MAY MOVE TO ANOTHER LAN
 - TABLE IS NOT NECESSARILY COMPLETE AT ANY POINT IN TIME.

LOOPS



• WHY WOULD THERE BE LOOPS?

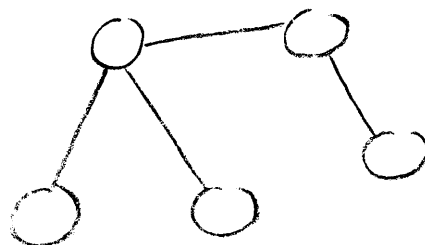
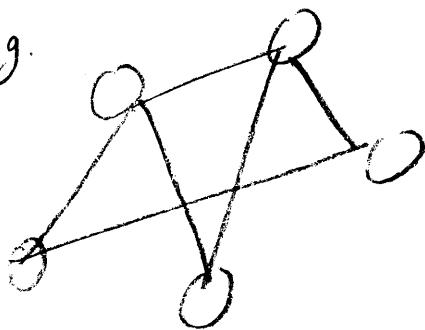
- COINCIDENCE, NETWORK MANAGED BY MORE THAN ONE PERSON, NO ONE REALLY KNOWS THERE IS A LOOP
- ON PURPOSE, TO PROVIDE REDUNDANCY IN CASE OF FAILURE

- SOLUTION: MUST BUILD A SPANNING TREE
- MODEL NETWORK AS GRAPH
 - < BRIDGES & LANS ARE NODES
 - < PORTS ARE EDGES

SPANNING TREE

- A SPANNING TREE IS AN ACYCLIC GRAPH (GRAPH CONTAINS NO CYCLES) THAT SPANS ALL THE NODES.

e.g.

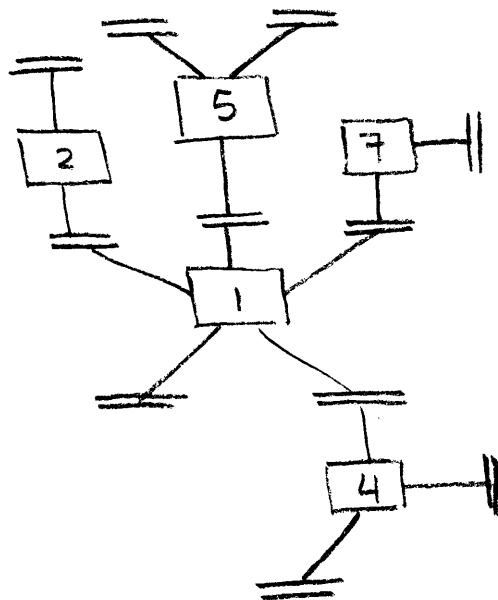


NO CYCLES

- IN OUR PROBLEM WE NEED TO SPAN ALL LANs (BUT NOT NECESSARILY ALL BRIDGES)

EXAMPLE

- A SPANNING TREE FOR THE PREVIOUS EXAMPLE

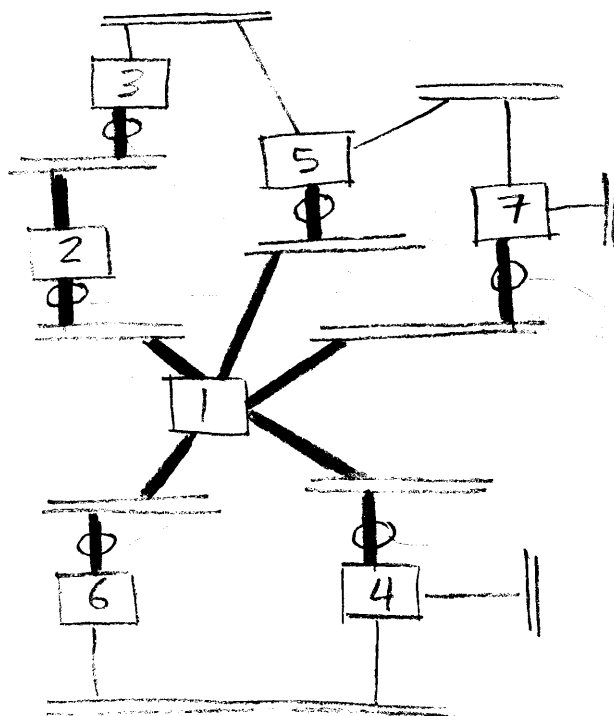


COMPUTING A SPANNING TREE

- ONE WAY TO COMPUTE A SPANNING TREE IS TO FIND A SHORTEST PATH TREE FROM A GIVEN SOURCE
 - (1) - CHOOSE A SOURCE NODE (BRIDGE)
 - (2) - EVERY BRIDGE COMPUTES THE SHORTEST PATH TO THE SOURCE, SETS PORT ON THAT PATH AS PREFERRED PORT
 - (3) - EVERY LAN SELECTS THE CLOSEST BRIDGE TO SOURCE (LANs ARE NOT ACTUALLY ACTIVE IN THIS PROCESS IT'S THE BRIDGES WHO ELECT ONE AMONG THEM)
- ALL TIES ARE BROKEN USING BRIDGES' INDICES TO FAVOR SMALLER INDEX
- SOURCE IS USUALLY BRIDGE WITH SMALLEST INDEX TOO

EXAMPLE

- SAY BRIDGE 1 IS THE SOURCE



- THICK PORTS SHOW THE SHORTEST PATH TO SOURCE FOR EVERY BRIDGE
- NOTE THAT BRIDGE 3 CHOOSES THE PATH THROUGH 2 RATHER THAN THE PATH THROUGH 5 (BOTH HAVE DISTANCE 2 TO SOURCE) BECAUSE SMALLER INDICES ARE FAVORED.
- NOTE THAT WE DID NOT SPAN ALL LANs YET (NEED STEP (3) ABOVE)

DISTRIBUTED ALGORITHM

- EVERY NODE (BRIDGE) X MAINTAINS
 - $X.s$: WHAT X THINKS THE SOURCE IS
 - $X.d$: DISTANCE FROM X TO SOURCE
 - $X.p$: PREFERRED PORT
 - $X.P$: SET OF ACTIVE PORTS
- NEIGHBORING NODES EXCHANGE MESSAGES OF THE FORM (Y, d, X)
 - I AM NODE X AND I AM AT DISTANCE d FROM SOURCE Y
 - EVERY NODE STARTS INITIALLY WITH $(X, 0, X)$
ie. CLAIMING TO BE THE SOURCE

DISTRIBUTED ALGORITHM

NODE X

UPON RECEIVING A MESSAGE (Y, d, Z) ON PORT P

if $(Y < X.s)$

then $X.s \leftarrow Y$

$X.d \leftarrow d + 1$

$X.P \leftarrow$ All ports

$X.p \leftarrow P$

if $(Y = X.s)$ AND $(d + 1 < X.d)$

then $X.d \leftarrow d + 1$

$X.p \leftarrow P$

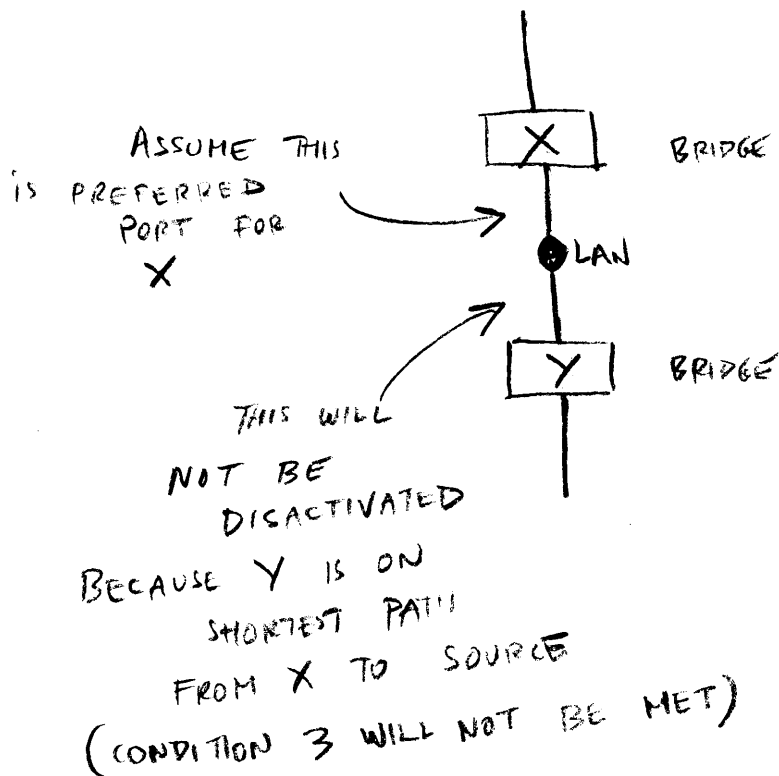
if $(Y = X.s)$ AND $[(d < X.d)$ OR $(d = X.d)$ AND $Z < X]$

then $X.P \leftarrow X.P - \{P\} \cup \{X.p\}$

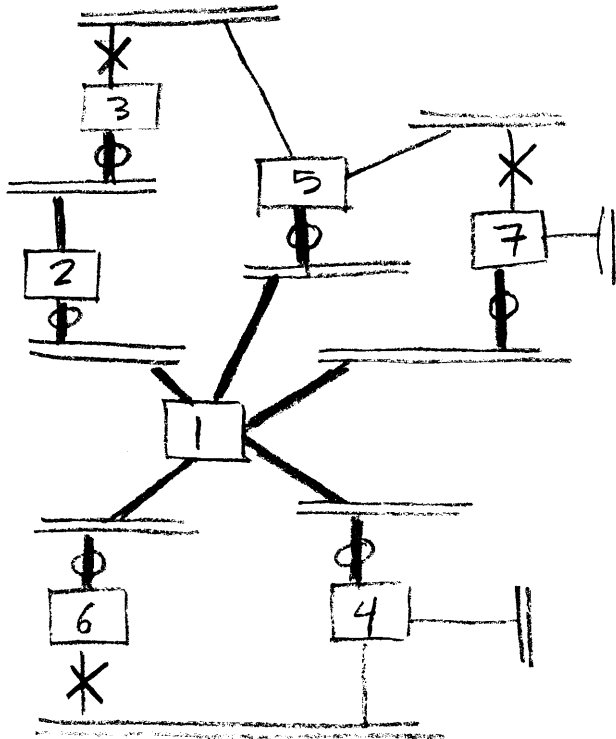
EXPLANATION

- NODE X UPDATES SOURCE WHEN IT SEES A SOURCE WITH SMALLER INDEX
 - IT REACTIVATES ALL ITS PORTS (NEW SOURCE)
 - IT RE COMPUTES ITS DISTANCE
 - IT SETS PREFERRED PORT
- NODE X UPDATES DISTANCE WHEN IT SEES BETTER PATH
 - IT ALSO UPDATES PREFERRED PORT (NEW PATH)
- NODE X DISACTIVATES A PORT WHEN IT SEES ANOTHER BRIDGE WITH SMALLER DISTANCE OR SAME DISTANCE & SMALLER INDEX (THIS IS EQUIVALENT TO STEP 3) UNLESS THE PORT IS THE PREFERRED PORT

ILLUSTRATION



RESULT



O: PREFERRED PORT
X: DISACTIVATED PORT

BRIDGES WITH JUST ONE ACTIVE PORT REMAINING, CAN DISACTIVATE THE PORT

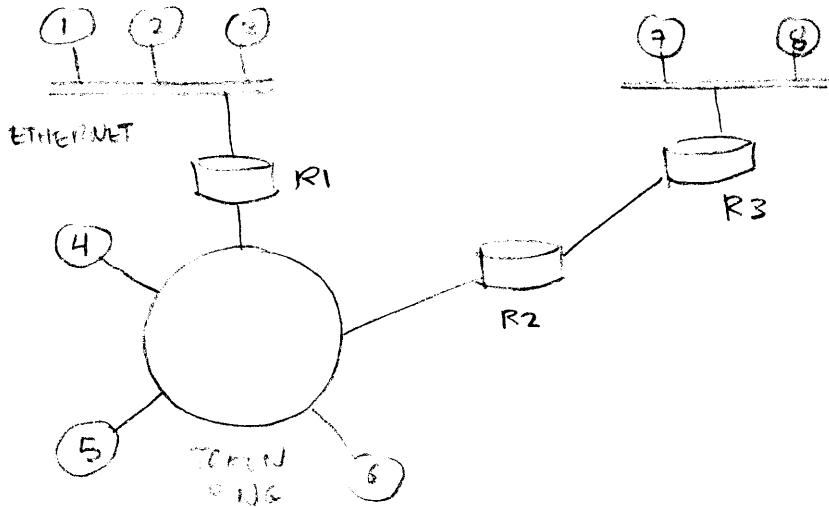
WE END UP WITH RESULT ILLUSTRATED AT THE BEGINNING.

LIMITATION OF BRIDGES

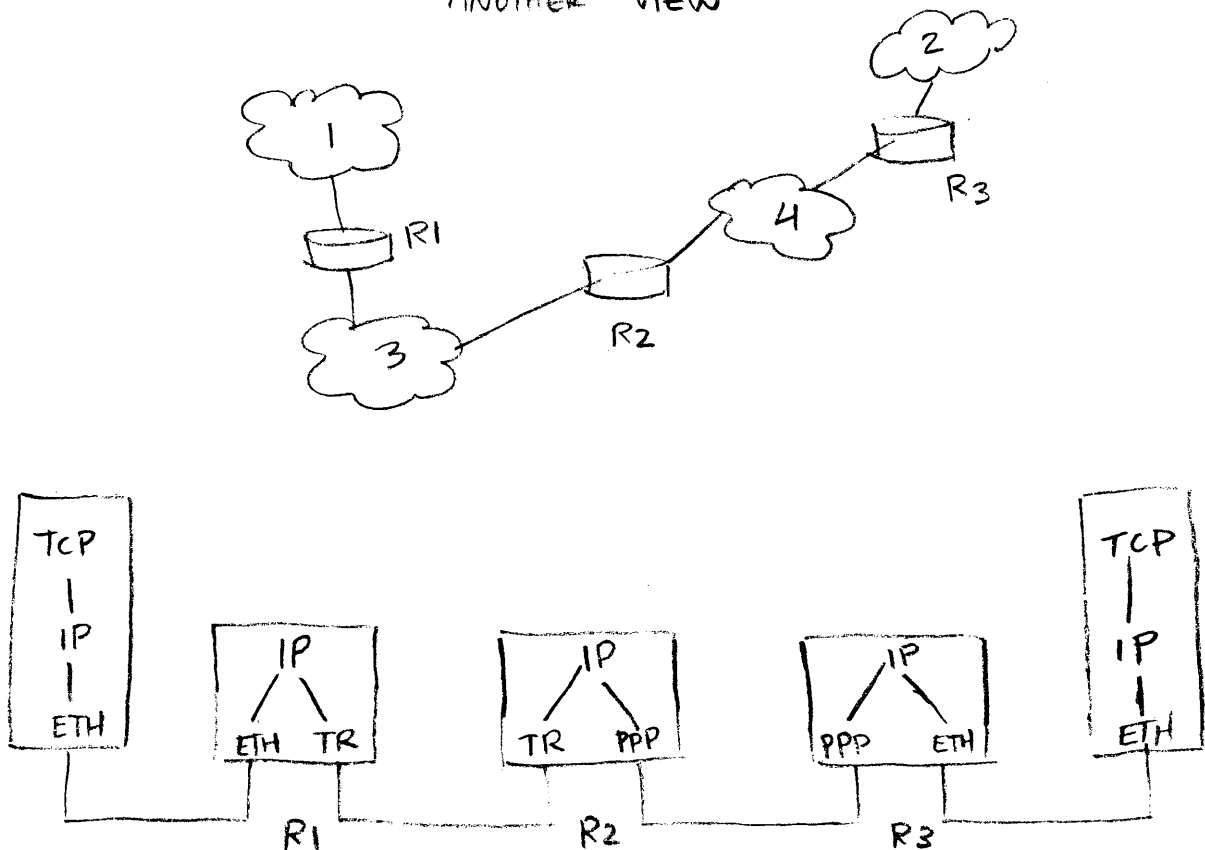
- NETWORKS MUST BE HOMOGENEOUS (eg. ETHERNET TO ETHERNET OR TOKEN RING TO TOKEN RING)
MUST HAVE SAME FRAME FORMAT
- BRIDGES MAY BECOME CONGESTED & DROP PACKETS
⇒ UNEXPECTED ON NETWORKS LIKE ETHERNET
- PRIVACY: BROADCASTING SOMETIMES NEED TO BE LIMITED.

BUILDING LARGER NETWORKS

- INTERNETWORK OR INTERNET : COLLECTION OF ARBITRARY NETWORKS CONNECTED TO PROVIDE HOST TO HOST SERVICE EACH NETWORK CAN BE DIFFERENT IN TECHNOLOGY / PROTOCOLS .
- THE NETWORKS ARE CONNECTED BY ROUTERS (OR GATEWAYS)



ANOTHER VIEW

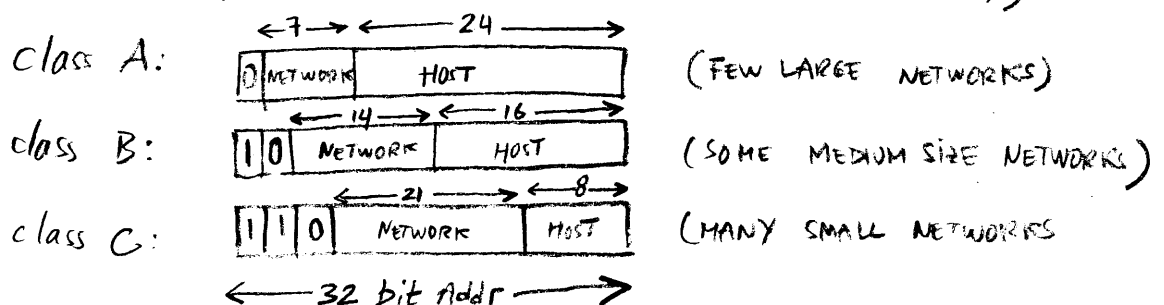


IP

- EACH ROUTER RUNS THE IP PROTOCOL
- IP SERVICE MODEL: DATAGRAM, CONNECTIONLESS, NO GUARANTEES, BEST EFFORT
- PHILOSOPHY: ANY ARBITRARY NETWORK CAN BE CONNECTED TO THE INTERNET (SCALABILITY)
e.g. CANNOT ENFORCE IP TO PROVIDE DELAY GUARANTEES BECAUSE A NETWORK THAT ARBITRARILY DELAYS PACKETS CAN BE ADDED
- EACH ROUTER HAS TO INTERFACE WITH EVERY NETWORK IT'S ON
- IP CONNECTIONLESS → EVERY PACKET CARRIES ENOUGH INFORMATION TO GET TO DESTINATION (IP ADDRESS)

IP ADDRESS

- IP ADDRESS IS A GLOBAL ADDRESS IN THE INTERNET (NOT THE SAME AS PHYSICAL ADDRESS ON NETWORK e.g. ETHERNET ADDRESS)
- EACH HOST HAS A PHYSICAL ADDRESS & IS ASSIGNED AN IP ADDRESS
- EACH ROUTER HAS A PHYSICAL & IP ADDRESSES / INTERFACE
- IP ADDRESSES ARE HIERARCHICAL (FOR SCALABILITY)



ROUTERS

- EACH ROUTER MAINTAINS A ROUTING TABLE

NETWORK #	NEXT HOP

- SCALABILITY IS ACHIEVED BY THE HIERARCHICAL NATURE OF IP ADDRESSING BECAUSE A ROUTER NEED ONLY MAINTAIN A LIST OF NETWORK # (V.S. A LIST OF ALL HOSTS!)

ROUTING

HOST: UPON RECEIVING A PACKET

if (NETWORK # OF DESTINATION = MY NETWORK #)

then DELIVER TO DESTINATION (C.G. ON ETHERNET)
(HERE A CONVERSION FROM IP TO PHYSICAL ADDRESS IS NEEDED, HOST MAINTAINS A TABLE THAT IT LEARNS OVER TIME)

else DELIVER TO DEFAULT ROUTER
(EVERY HOST IS CONFIGURED WITH A DEFAULT ROUTER SITTING ON THE NETWORK, C.G. THE GATEWAY)