Topology Discovery for Large Ethernet Networks

HAW Hamburg
LIU, Chang

1. Introduction

- Both network management and performance analysis benefit from network topology knowledge.
  - Network managers are better able to react to and prevent problems
  - The complexity of performing Ethernet topology discovery arises from the inherent transparency of Ethernet bridge hardware.

- A new algorithm that can perform automatic topology discovery without requiring complete knowledge:
  - Requires only forwarding entries for three machines to be shared between each pair of bridges.
  - With up to 2000 nodes and 50 bridges while using only one machine to send queries and pings.

2. Topology Discovery with Direct Connections

- Design an algorithm that determines the physical topology of the Ethernet network used to connect a set of hosts.
- Derive the network topology using information obtained from the bridges themselves with SNMP

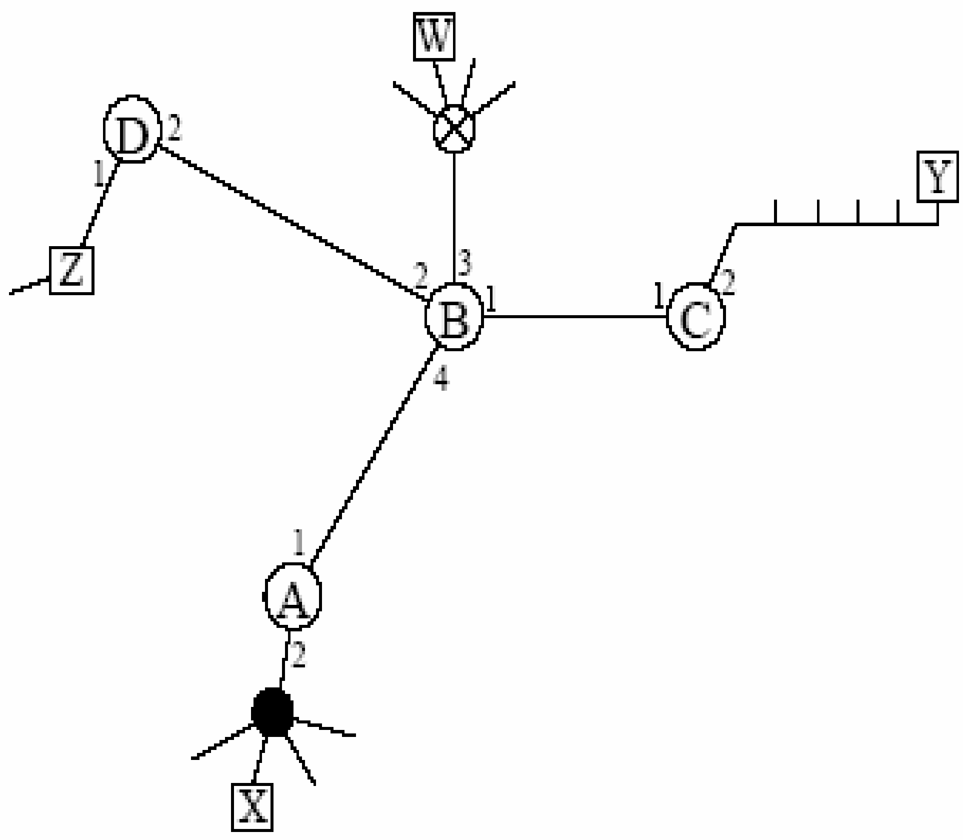
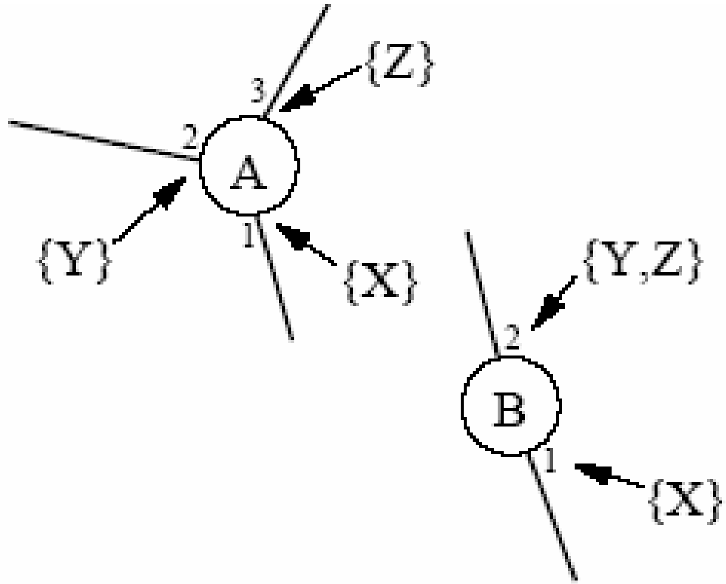


Figure1: A sample bridged Ethernet network.

- A, B, C, and D are bridges.
- W, X, and Y are hosts.
- Z is a router.
- The port numbers of the bridges are indicated next to their respective links.
- A hub is shown connected to A, a non-SNMP enabled switch to B, and a shared segment to C.

3. Topology Discovery with Incomplete Knowledge

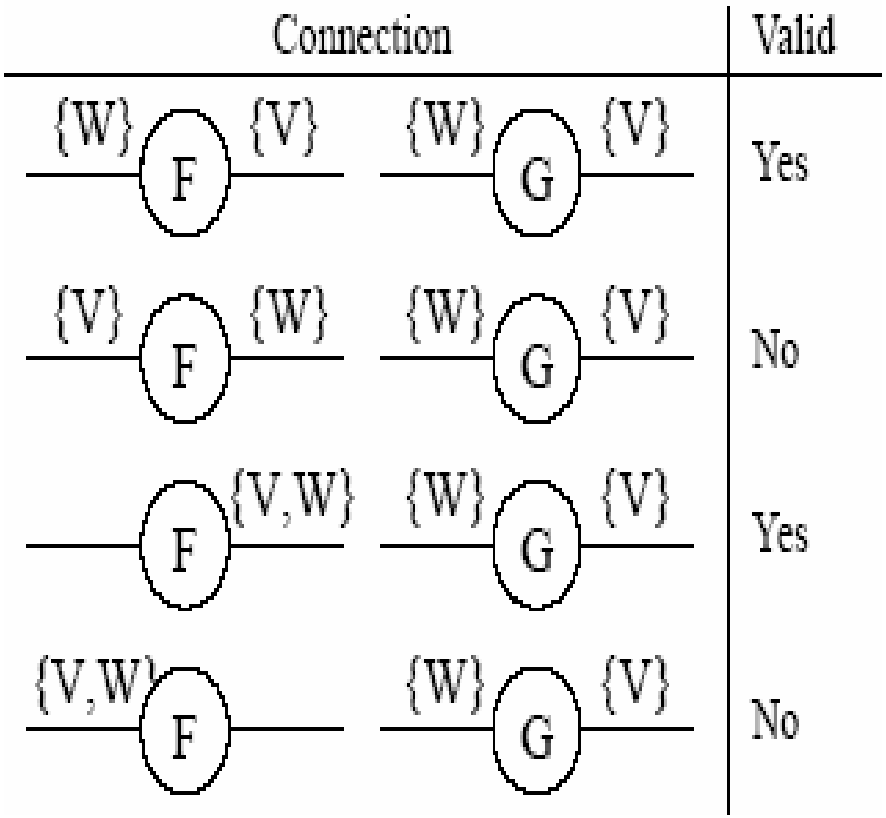
- Difficulties associated with obtaining complete forwarding databases for entire Ethernet LANs
- Alternative approach to topology discovery:
  - Rather than proceeding with the goal to prove that two nodes are directly connected, consider the approach of proving that bridge a is not simply connected to port x of b, denoted bx.
  - Because this is a proof by contradiction, only a single counterexample is needed to demonstrate the contradiction.



Bridge	Port	Forwarding Entries	Through Sets
A	1	X	Y, Z
	2	Y	X, Z
	3	Z	X, Y
B	1	X	Y, Z
	2	Y, Z	X

- Two bridges are simply connected by a particular pair of ports.
- There may be nodes in between- > not a direct connection
- If all entries in the FDBs are consistent with this connection- > it may exist
- If any one entry is inconsistent- > connection is impossible.

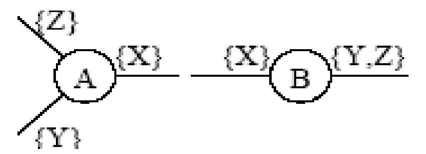
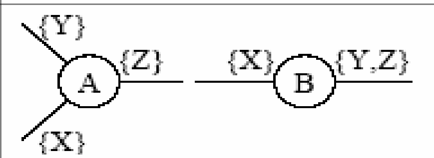
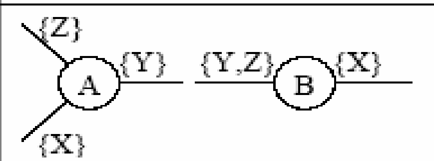
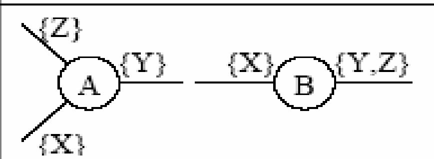
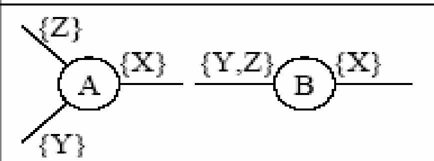
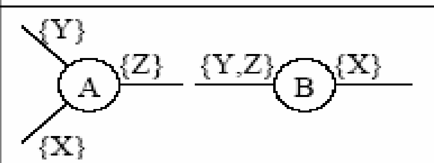
Figure 2: Example of two bridges for which contradictions can be used to determine the connections.



- In each case, the forwarding entries are examined to see whether those two edges can form a valid simple connection.

- Note that the only cases where there are conflicts occur when the bridges map the same address in opposite directions.

Figure 3: Examples of valid and invalid connections between two bridges with different forwarding entries on bridge F.

Ports A	Ports B	Mapping	Conflict
1	1		Y and Z
3	1		Y
2	2		X
2	1		Z
1	2		NONE
3	2		X

- After determining through sets for each port, it is simple to apply this information to determine the ports that connect two bridges.

- see if ports x and y of bridges a and b are connected, calculate $T_a^x \cap T_b^y$. If there are any addresses in common, then the ports cannot be connected

Figure 4: How contradictions can be used to eliminate impossible connections from the bridges in Figure 2.

3. Rigorous presentation

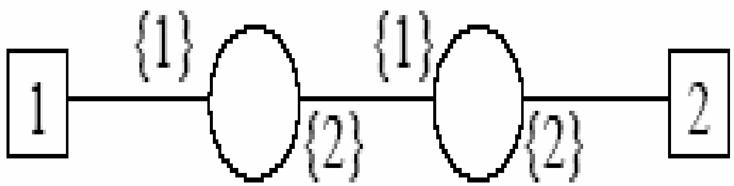


Figure 5

- This network consists of two bridges and two hosts, labeled 1 and 2.
- The connection of the two bridges cannot be determined because there is insufficient information.
- Observe that the bridges can be placed in either position, as they are indistinguishable using their forwarding databases.

Examples of the three required shared entries of the Minimum Knowledge Requirement

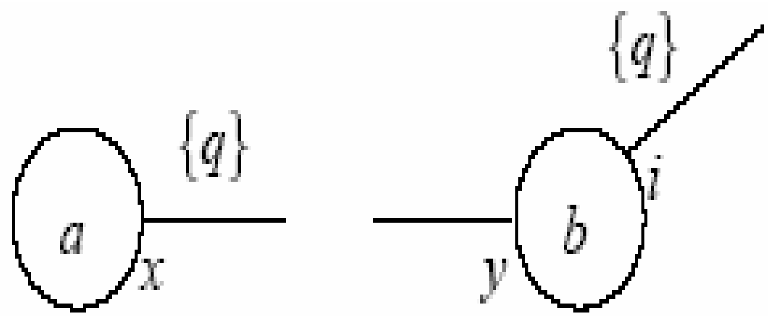


Figure 6(a): illustrates a shared entry between *ax* and *bi*, which is insufficient to determine *x*.

- *ax* can be connected to any port on *b* without creating a conflict.
- *bi* can also be connected to any port on *a* without creating a conflict
- *q* will be forwarded from *b* through *bi* to *a* and on through *ax*.
- Because *bi* can be connected to any port on *a*, this single shared entry does not uniquely define *x*.

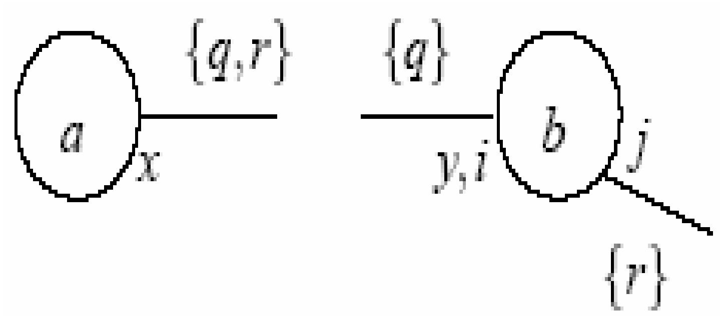


Figure 6(b): adds a shared entry with *b j* , which determines *x*, but not *y*. Note that although *i =/= j*, *y* can equal *i* or *j*.

- **if *ax* is connected to *b*, the connection is valid. However, if any port of *a* other than *ax* is used for the connection, a conflict is created.**
- **If *bi* is connected to a port on *a* other than *ax*, *r* will create a conflict because it is forwarded in opposite directions.**
- **Likewise, a conflict will be created if *bj* is connected to a port other than *ax*, as *q* will now be forwarded in opposite directions. If a port on *b* other than bi or bj is connected to a port other than ax, then both q and r will present conflicts.**
- **Therefore, we see that these two shared entries uniquely determine ax as the port used to connect a to b, although y is still not determined, as any port of b can be connected to ax without generating a conflict.**

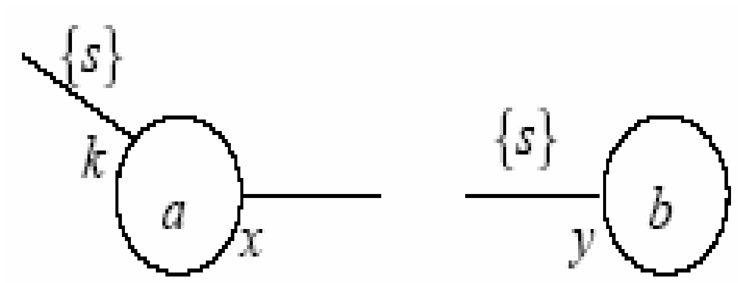
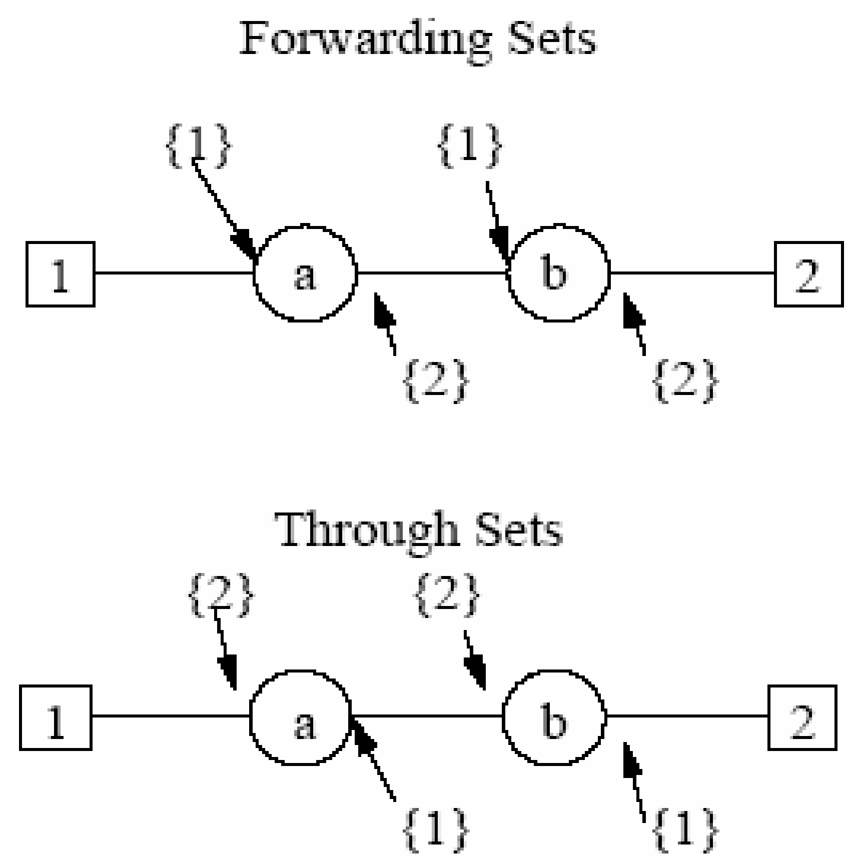


Figure 6(c): illustrates the shared entry needed to determine y once x is determined. Note that k =/= x.

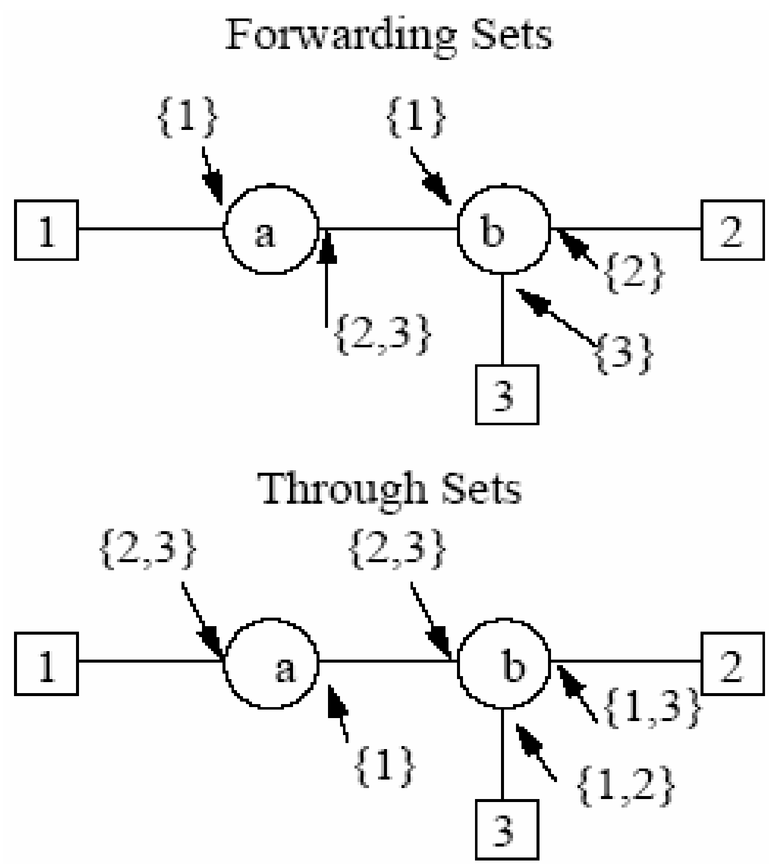
- **Rule: Once x is uniquely determined, y is determined with the same rule as used for condition two.**
- **Because k =/= x, the shared entry s between *ak* and *by* forces y to be the port connected to a, otherwise a conflict would be creating by mapping s in opposite directions.**

Two network graphs are shown with forwarding and through sets indicated:



- There is not a unique solution to the mapping problem.
- The bridges share two entries, but they are symmetric and can be placed in either order.

Figure 7: Network (a) does not provide sufficient information to meet the minimum knowledge requirement.



- Resolves the indeterminism by adding a second machine off of one of the bridges which is shared in both bridges' FDBs, the network is uniquely determined.
- This second machine gives *a* shared entries with two of *b*'s ports. It is no longer possible to reorder these two bridges because the only port on the left bridge that can be connected to the right one is the one with both entries.
- Any other connection will cause a contradiction.

Figure 7: Network (b) however, with one additional host, meets that requirement.

4. Practicality Example

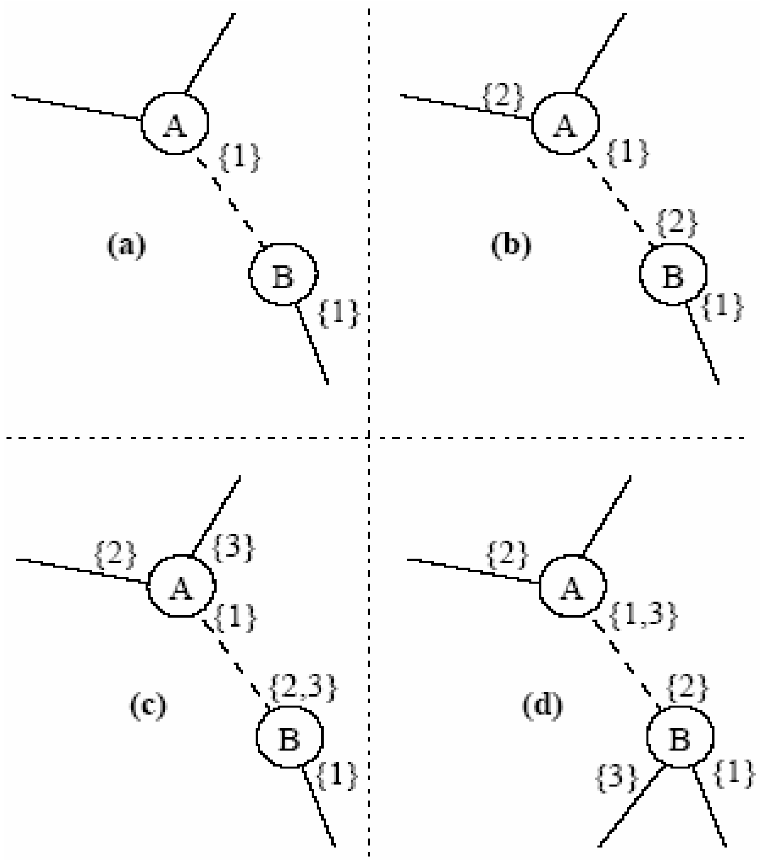


Figure 8

- These snapshots of two bridges within a network demonstrate different examples of indeterminate and determinate FDBs.
- Imagine that the real position of these two bridges has bridge A internal to the topology and bridge B positioned as a leaf bridge, connecting only to one other bridge, A, with the remainder of its ports connected to endpoints.
- The dashed line is the physical connection that the algorithm is attempting to determine.

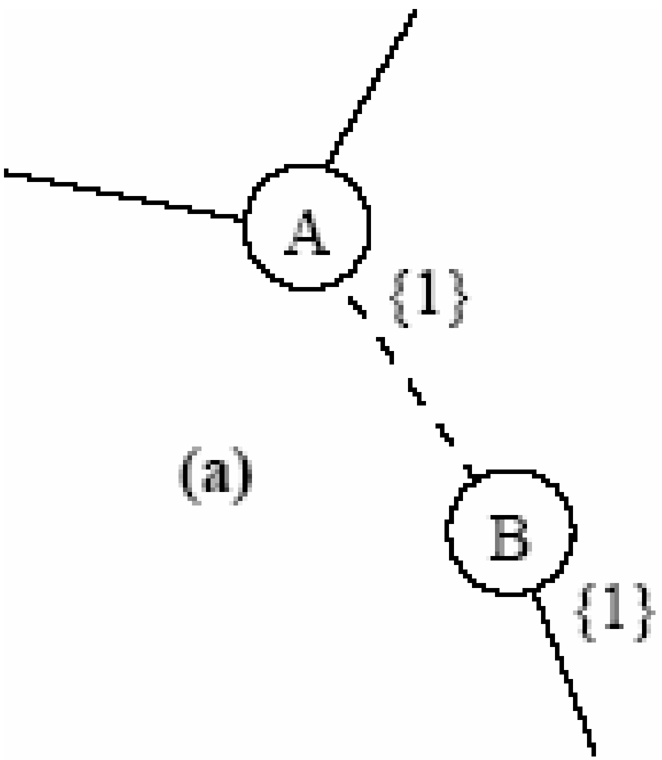


Figure 8 (a)

- Only one host has an entry shared between the two bridges.
- This mapping is obviously indeterminate.

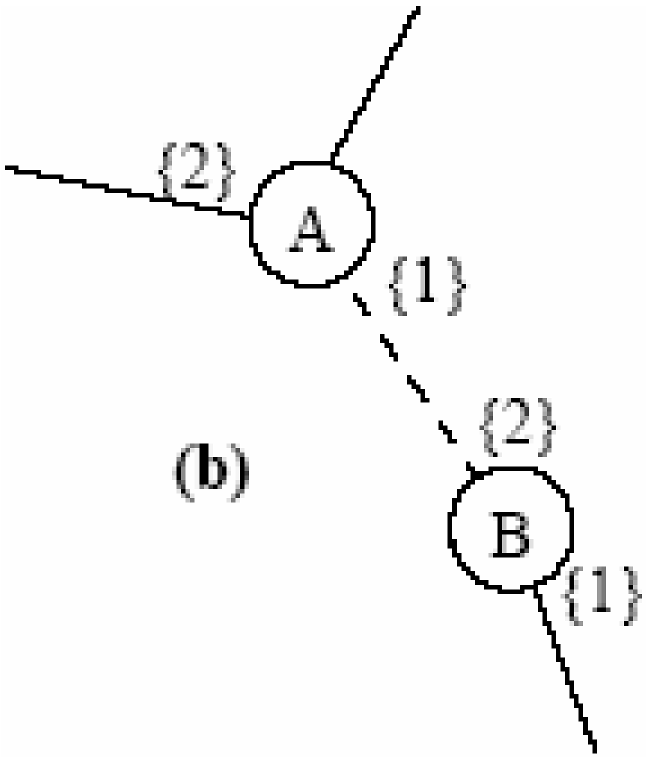


Figure 8 (b)

- The new node 2 might correspond to the querying node sending pings to node 1 while probing the FDBs.
- However, for the same reasons as Figure 5, this mapping is still indeterminate.

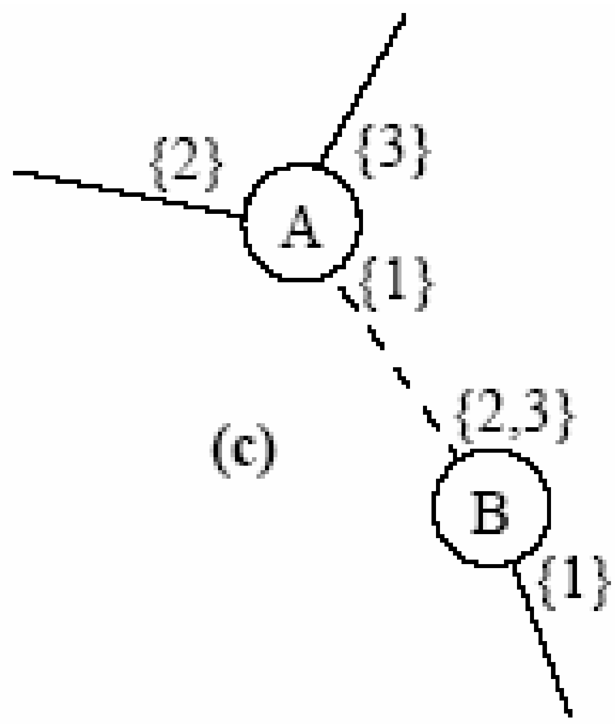
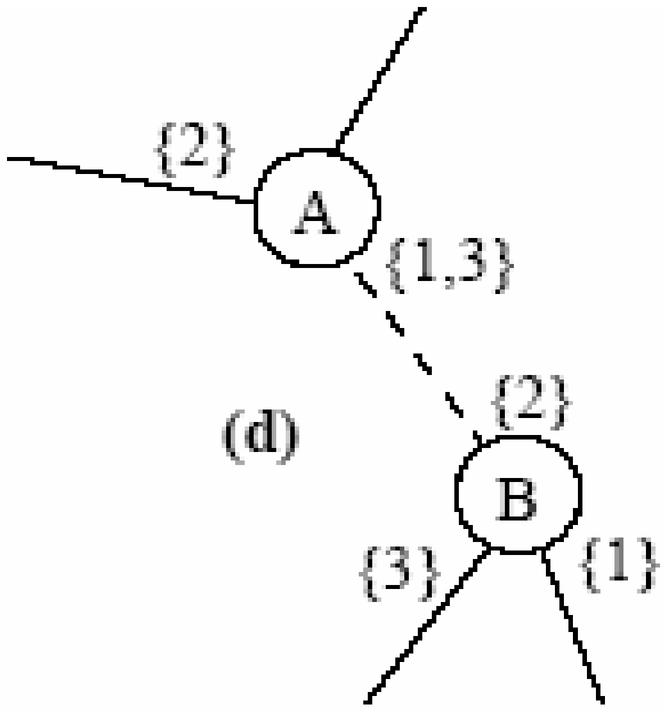


Figure 8 (c)

- Suppose that node 1 has communicated with some other host on the network, for instance a nameserver.
- If that node is also found on any port of the upper bridge other than the port with 2 in its forwarding set, then the mapping is determinate.
- In many cases, it is possible to force a machine to contact a nameserver by connecting to its FTP daemon, for instance.



- In most cases, networks aren't designed with only one endpoint connected to a bridge. Almost all bridges have 4, 16, 24, or even more ports used for connections to machines.
- The only entries in the FDBs are for endpoints on two different ports of bridge B and for the querying machine sending them pings.

Figure 8 (d): A bridge with just two ports connecting machines to the network

- Networks (c) and (d) are two minimal examples:
  - a single machine on a bridge that has talked to more than one other machine (host, server, or router)
  - or a bridge connected to two machines.

- Practically every part of an Ethernet will meet at least one of these criteria.

- However, because bridges are generally only installed in a network to be used, which means there are machines divided among several ports, most components will provide information well beyond the minimal requirements.

5.1 Implementation

- Two situations where a direct connection could not be established between two bridges in an otherwise complete network.

- The principal challenge is the creation of a shared Ethernet segment, rather than the modern point-to-point connections.

  - A hub is used to connect two bridges with other hosts or bridges. Because hubs do not participate in the bridging algorithm, this creates a shared network segment between the bridges.

  - A bridge exists that the algorithm either was not informed about, or to which SNMP access is denied. Because SNMP security generally consists of a simple list of allowed or denied IP addresses, this situation can easily occur.

5.2 Virtual Switches

- Rather than requiring a special case, virtual switches are naturally determined using the base algorithm.

- In fact, the intersection set is calculated as the algorithm searches for the next bridge, so when no directly connected bridge is found, the nodes involved in the shared segment are already determined.

6. Reference

- Topology Discovery for Large Ethernet Networks
  - Bruce Lowekamp
  - Computer Science Dept. College of William and Mary Williamsburg, VA
  - lowekamp@cs.wm.edu

QUESTION ???

- END -