

CSE 770: Network Seminar

Ethernet Topology Discovery without Network Assistance

Richard Black, Austin Donnelly, Cédric Fournet

Presenter: Mike Wilson

Discussion Leader:
Charlie Wiseman

(Not just another pair of Germans!)



 Washington University in St. Louis

The Problem

The most costly help desk calls are those related to local area network (LAN) problems

- Initial and slowest step is usually to determine the topology
- Most LAN environments do not have a dedicated support staff
- Enterprise networks usually have high-end management tools (SNMP, MIBs). LANs do not
- Topology knowledge is also necessary for bandwidth allocation problems

2 - Ethernet Topology Discovery without Network Assistance

Mike Wilson

 Washington University in St. Louis

Solutions

- **Internet-scale tomography**
 - Usually IP-layer, does not apply well to the LAN
- **Enterprise Network mapping**
 - Requires enterprise equipment using SNMP
 - Problems with heterogeneous implementations
 - SNMP is not supported on wireless nodes

Ethernet Discovery

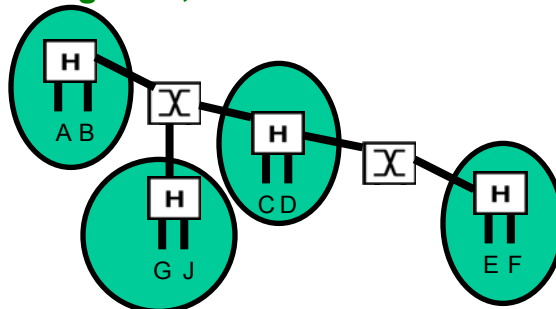
- Truly useful solutions must not make assumptions about the network elements**
- **Switches vary greatly in their SNMP capabilities**
 - **Switches vary greatly in adherence to the Ethernet standard**
 - **Switch performance varies greatly**

Ethernet Discovery

- The authors' algorithm relies only on observable phenomena, not queries.
- Probe packets are sent, and the topology is inferred from which hosts receive these packets
- Host capabilities are limited to sending packets with arbitrary MAC addresses and sniffing promiscuously

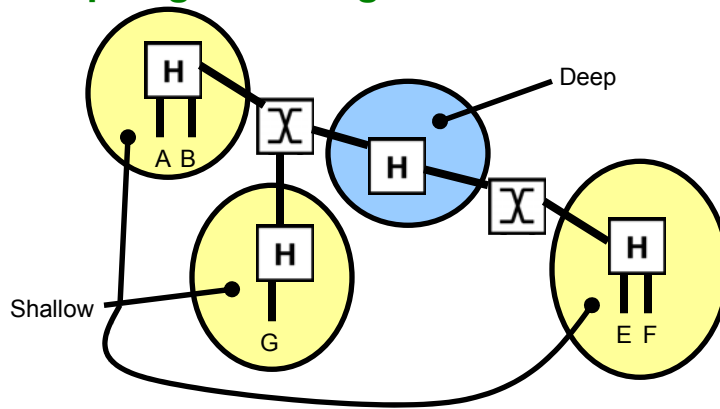
Algorithm Definitions

- **Segment:** a portion of an Ethernet where all members can overhear all transmissions. Normally, this consists of a hub or hubs, all daisy-chained together. While a switch may be a member of a segment, the segment boundary does not cross the switch. Note: technically each cable is a segment, as the base case.



Algorithm Definitions

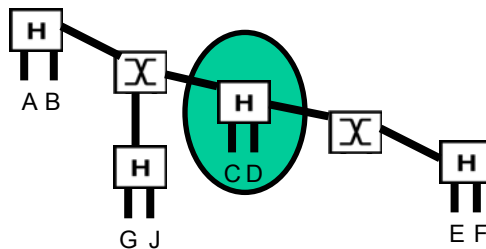
- **Shallow segment:** segment with at least one host.
- **Deep segment:** segment with no host.



7 - Ethernet Topology Discovery without Network Assistance Mike Wilson Washington University in St. Louis

Algorithm Definitions

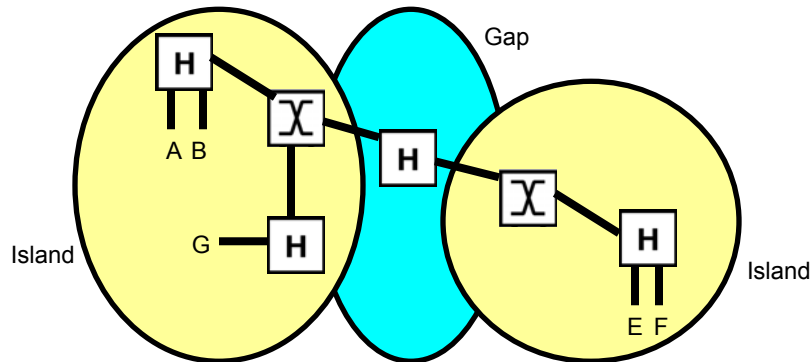
- **Intermediate segment:** segment where a packet transmission is heard by at least two switches. That is, there are at least two switches and a third entity which can send packets (switch or host) all connected by shared media (hubs).



8 - Ethernet Topology Discovery without Network Assistance Mike Wilson Washington University in St. Louis

Algorithm Definitions

- **Island:** one or more shallow segments, forming a maximal connected subgraph with no intervening deep segments. That is, shallow segments that are connected by any construct not involving a switch-to-switch segment that has no hosts on it.
- **Gap:** portion of a network made up solely of deep segments.



9 - Ethernet Topology Discovery without Network Assistance Mike Wilson Washington University in St. Louis

Algorithm Notation

- Some of the techniques used rely on forging ethernet frame source addresses. The authors use $A : X \rightarrow B$ to indicate that the host with MAC address A sends a frame to MAC address B with source address X .
- Some of the techniques rely on having a special “per-host” address L . It is not always obvious that when the authors indicate $A : L \rightarrow X$ followed by $B : L \rightarrow X$, that these are two *different* addresses L_A and L_B .

10 - Ethernet Topology Discovery without Network Assistance Mike Wilson Washington University in St. Louis

Principles and Techniques

Entire algorithm is based on two concepts:

- Any host on a segment can detect all transmissions on that segment, even when these transmissions are directed elsewhere.
- A switch that hears a packet sourced from a given address will establish a routing to the port on which it heard that address. From this, we can develop a technique where a switch is presented a packet to establish a specific routing, and then another test packet is sent from another host using the same source address. If the switch sees the second packet, it will update the routing again. Finally, a third packet is sent to test the state of the routing table.

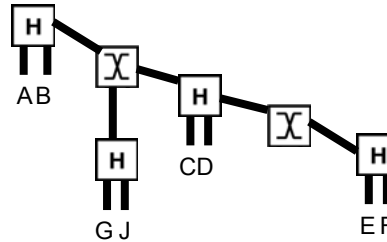
The Algorithm: Overview

4 Phases:

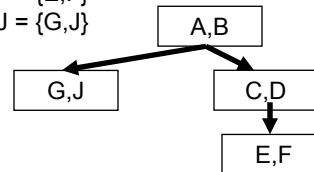
1. Segment Detection
(All segments discovered, segment tree sorted)
2. Shared Switch Detection
(Locates switches shared between segments.
All islands are fully mapped.)
3. Island Edge Determination
(Locates all gaps between each island.)
4. Gap Topology Determination
(Determines topology of each gap to simplest point of observational equivalence.)

Phase 1: Segment Detection

1. Select a collector
2. All hosts promiscuous
3. All hosts send probe to collector
4. Collector also sends probe to it's own L
5. For each host, determine "sees" set. All hosts with identical "sees" sets are in the same segment
6. Organize segments into an overhearing hierarchy (segment tree)
7. Elect segment leaders



"Sees" sets:
 A,B = {A,B,C,D,E,F,G,J}
 C,D = {C,D,E,F}
 E,F = {E,F}
 G,J = {G,J}

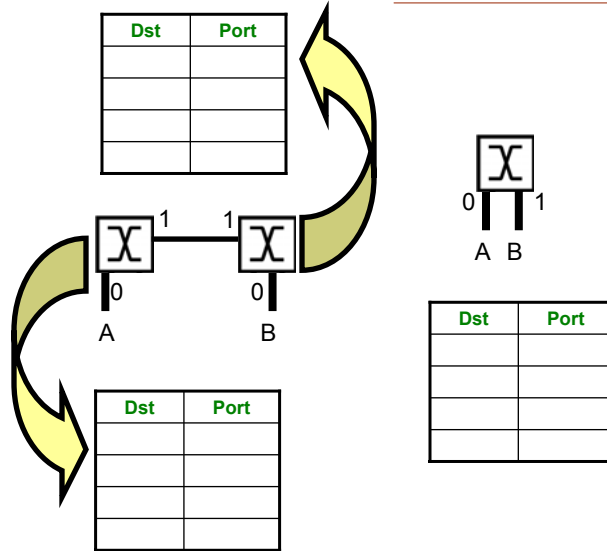


Phase 2: Shared Switch Detection

Using the switch training technique, detect switches shared directly between segments.

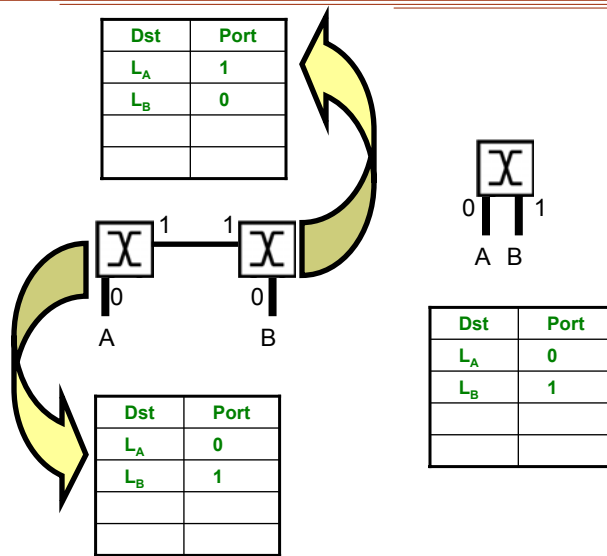


Training Technique



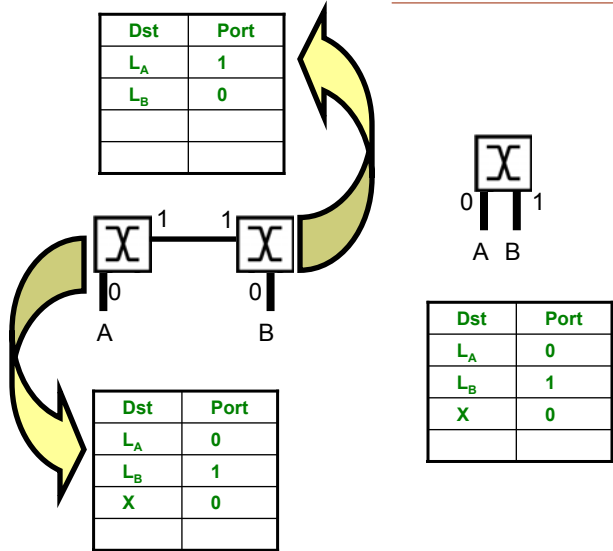
Training Technique

1. A : $L_A \rightarrow B$
2. B : $L_B \rightarrow A$



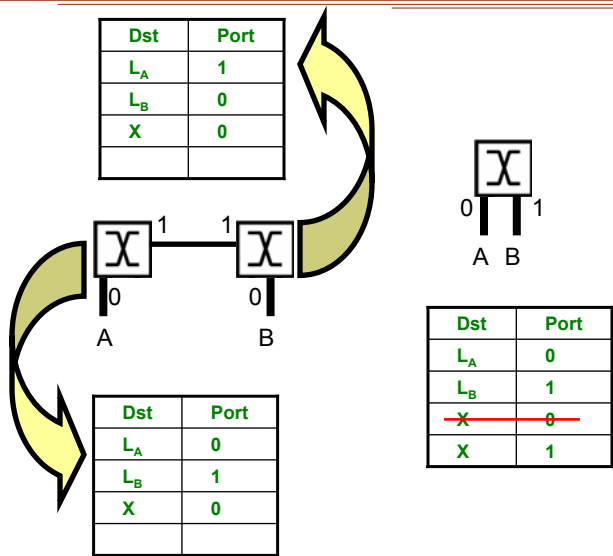
Training Technique

1. $A : L_A \rightarrow B$
2. $B : L_B \rightarrow A$
3. $A : X \rightarrow L_A$



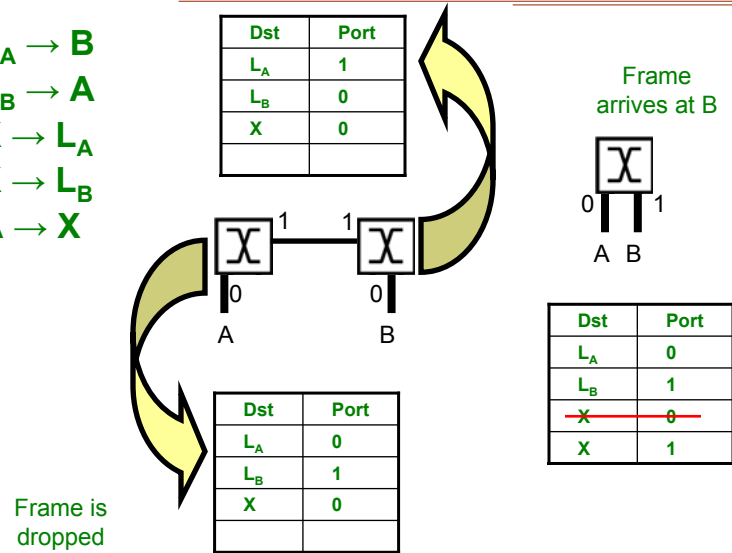
Training Technique

1. $A : L_A \rightarrow B$
2. $B : L_B \rightarrow A$
3. $A : X \rightarrow L_A$
4. $B : X \rightarrow L_B$



Training Technique

1. $A : L_A \rightarrow B$
2. $B : L_B \rightarrow A$
3. $A : X \rightarrow L_A$
4. $B : X \rightarrow L_B$
5. $A : A \rightarrow X$



19 - Ethernet Topology Discovery without Network Assistance Mike Wilson Washington University in St. Louis

General Process for Phase 2

1. Using the segment tree, process segment leaders in pre-order depth-first traverse.
 1. Each segment leader sends a training frame, $S_i : X \rightarrow L$.
2. Each segment leader sends a probe $S_i : S_i \rightarrow X$
3. All segment leaders listen for the probes.
4. If S_i hears probes from S_j, S_k, S_m this means that segments i, j, k and m share a single switch.

20 - Ethernet Topology Discovery without Network Assistance Mike Wilson Washington University in St. Louis

Results of Phase 2

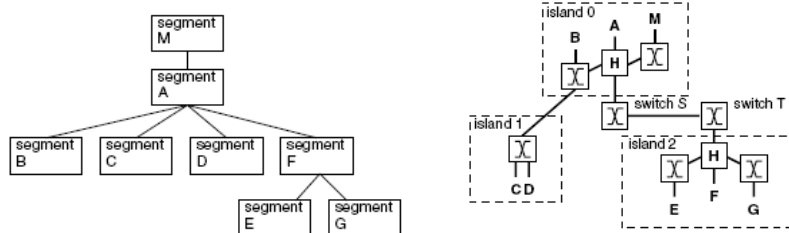
At the end of phase two we have discovered:

- All shallow segments, and
- All islands, and
- A hierarchy of segments

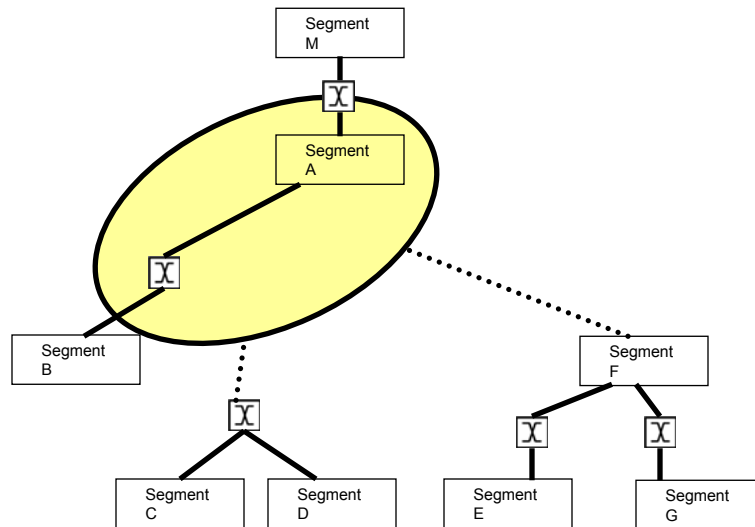
Note that we already have a complete topology for each island!

Phase 3: Island Edge Determination

The segment tree tells us which islands are connected to each other. What we don't know is where the connection points are.



Phase 3: Island Edge Determination



23 - Ethernet Topology Discovery without Network Assistance Mike Wilson Washington University in St. Louis

Phase 3: Island Edge Determination

- 1. For each parent-child island gap:**
 1. If the parent island's attachment segment has no child switches, infer one. The child island attaches through this switch.
 2. Otherwise, for each child switch of the parent island:
 1. Send a packet from the child island to a segment below the candidate switch. If the parent segment of the switch fails to see the packet, then the child island attaches through this switch.
 3. If no attachment switch is found, infer a new one. The child attaches through this switch.
 4. If the child island has a parent switch, the attachment is through this switch.
 5. Otherwise, we infer the existence of a parent switch. The attachment goes through this switch.

24 - Ethernet Topology Discovery without Network Assistance Mike Wilson Washington University in St. Louis

Phase 3: Island Edge Determination

For the example:

1. Segment C sends to Segment B. Segment A does not see the packet. Therefore, Island 1 attaches to Island 0 through the switch between Segment A and Segment B.
2. Since Island 1 has a parent switch, the child-side of the attachment passes through this switch.
3. Segment F sends to Segment B. Segment A sees the packet. There being no other candidate switches, we infer a new switch under Segment A and attach here.
4. Since Island 2 has no parent switch, we infer one. The attachment passes through this switch.

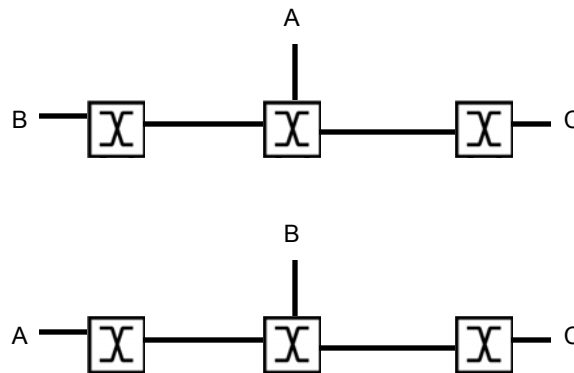
Results of Phase 3

- All Island attachment points are known.
- All two-switch gaps are known, but...
- For some island relationships, multiple islands attach to the same island segment. We don't yet know how these attachments share paths through the gap.

- After Phase 3 is complete, we elect a *Switch Leader* for each island bordering an unmapped gap.

Results of Phase 3

A was the collector. We know B and C are attached to an inferred switch below A. Which switch bridges the gap? This is a simple 3-switch gap.



Phase 4: Gap Topology Determination

Gap topology determination uses a *Path Crossing Test*, the AB/PQ test.

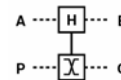
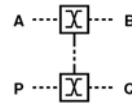
Given 4 switch leaders, A, B, P, and Q all in (different) islands bordering the same gap, the AB/PQ test determines how the path from A to B intersects the path from P to Q.

AB/PQ test

1. $A : X \rightarrow B$
2. $P : X \rightarrow Q$
3. $B : B \rightarrow X$

Results:

- Only P observes the probe.
- Only A observes the probe.
- Both A and P observe the probe.

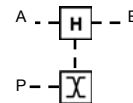
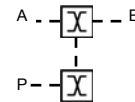
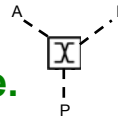


AB/PP test With 3 Islands

1. $A : X \rightarrow B$
2. $P : X \rightarrow L_p$
3. $B : B \rightarrow X$

Results:

- Only P observes the probe.
- Only A observes the probe.
- Both A and P observe the probe.



Phase 4: Algorithm

1. **For each switch leader, test AB/PP with that switch as P and all other possible pairs as A,B.**
 1. If P observes the probe, we can split the current gap into several smaller gaps with fewer border switches. That is, P splits the gap.
2. **When the segment bordering a gap is an intermediate segment, then we can still train the switches of that segment by selecting P,Q such that the path PQ runs through this segment.**
 1. Apply AB/PQ recursively while this condition holds, splitting the gap until the condition no longer holds.

Phase 4: Algorithm (Cont'd)

3. **General case: Every edge to a gap of three or more switches is of the form:**



1. Choose P, Q so that PQ crosses the gap.
2. Group remaining switch leaders into equivalence classes based on AB/PQ.
3. Order the classes along PQ by sorting with AP/BQ. Yields number of switches along PQ with attachments, the number of attachments, and the switch leaders for each attachment.
4. Within each equivalence class, select a new Q and recursively applying step 3 of the algorithm.

Limitations

- The algorithm only finds topologies that are *observationally equivalent*. For most applications, this suffices.
- The algorithm fails in the presence of 802.1X port-based access control. This only allows specified MAC addresses to send traffic on a port. The algorithm relies on being able to send packets with arbitrary addresses.

Limitations (Cont'd)

- The direct algorithm fails with wireless access points and bridges, since these modify the MAC addresses in transit. Additionally, some wireless hosts have problems with promiscuous mode.
 - We can still detect wireless segments by using the BSSID; this happens at Phase 1.
 - We elect a segment leader for each AP or bridge during Phase 2 as normal. All techniques except training still work (we can't forge MAC addresses). However, APs are never intermediate segments, so this isn't a major problem. Likewise, a bridge is intermediate, but we can just map each side normally and then splice together the result.

Limitations (Cont'd)

- **Some hosts lack the capabilities send arbitrary ethernet frames (e.g., a networked printer). These are handled by detecting the hosts via passive monitoring, then using 3rd-party ARP to cause these hosts to send frames to arbitrary destinations. (accuracy is reduced.)**
- **Many switches do not fully adhere to 802.1D, the relevant portion of the Ethernet Standard.**

Limitations (Non-compliant Switches)

- **802.1D defines a range of addresses that should not be propagated, used by STP. Consumer-grade switches propagate these anyway in an attempt to be invisible to STP. Workaround: use switch training and a special range of MAC addresses.**
- **Enterprise-grade switches have higher latency (150 ms) on learning new routings than consumer grade switches. Solution: wait 150 ms after each learning sequence.**

Limitations (Non-compliant Switches)

- **Conexant CX84200 chipset-based switches have a major defect: sometimes they “reflect” frames back out the port on which they arrived. Workaround: Algorithm has a special sequence to detect these chipsets and disregard the reflections.**
- **Linksys BEFW11S4 sometimes reflects frames back out the port on which they arrived. No workaround is yet available, since the authors do not have a reliable way to detect this switch.**
- **Not mentioned by the authors are layer-2 firewalls, which would also cause problems.**

Experimental Results

- **Implementation is about 4000 lines for the Mapper; 500 for the daemon.**

Topology	pkts	secs
(switch A)	6	1.10
(switch A B)	32	2.14
(switch A (switch B))	38	2.13
(switch A (AP B (bridge C)))	37	3.30
(switch A B (switch C D))	69	2.61
(switch A B (hub C D (switch E F)))	87	2.64
Three-switch problem (figure 7)	92	3.38
Figure 5	149	4.46

Experimental Results (Cont'd)

- **Algorithm complexity can be computed by phase. The authors note that the dominant factor is the 150 ms wait between training packets.**
 - Phase 1: $O(N)$, N = number of hosts
 - Phase 2: $O(S)$, S = number of switches
 - Phase 3: $O(ig^2)$, I = number of islands, g = largest number of gaps attached to any one island.
 - Phase 4: Authors do not provide a worst-case complexity value. They correctly note that this phase is rarely applied even in very complex networks. A back-of-envelope calculation shows an upper-bound of $O(N^4)$ where N = number of hosts, although this only applies in very unrealistic, pathological cases.

The Proof

- The authors offer a proof of the correctness of their algorithm. In very high-level overview:
1. Phase 1 correctness is demonstrated by an observability lemma.
 2. Phase 2 follows from ordering properties generated in phase 1.
 3. Phase 3 follows from ordering properties extended from phase 2.
 4. Phase 4 is demonstrated by a structural induction on the AB/PQ test applied to gaps of arbitrary size.

Questions?

**Moving along to our discussion leader,
Charlie Wiseman...**