# A Physical Topology Discovery Method Based on AFTs of Down Constraint

Bin Zhang[1], Xingchun Diao[2], Donghong Qin[3], Yi Liu[4], and Yun Yu[2]
[1]Cyberspace Security Research Center, Pengcheng Laboratory, China
[2]Nanjing Telecommunication Technology Research Institute, China
[3]School of Information Science and Engineering, GuangXi University for Nationalities, China
[4]National Innovation Institute of Defense Technology, Beijing, China

**Abstract**: *Network physical topology discovery is the key issue for network management and application, the physical topology discovery based on Address Forwarding Table (AFT) is a hot topic on current study. This paper defines three constraints of AFTs, and proposes a tree chopping algorithm based on AFTs satisfying down constraint, which can discover the physical topology of a subnet accurately. The proposed algorithm decreases the demand for AFT integrity dramatically, and is the loosest constraint for discovering physical topology which just relies on AFTs of down ports. The proposed algorithm can also be used in the switch domain of multiple subnets.*

**Keywords:** *Physical topology discovery, address forwarding table, network management.*

*Received January 27, 2015; accept September 9, 2015*

## 1. Introduction

Physical network topology refers to the characterization of the physical connectivity relationships that exist among entities in a network. Many network management tasks rely on knowledge of network physical connectivity. However, obtaining such information is a very difficult task. The majority of commercial network-management tools feature an IP mapping functionality for automatically discovering routers and subnets and generating a network layer topology showing the router-to-router interconnections and router interface-to-subnet relationships [19]. Unfortunately, layer-3 topology covers only a small fraction of the interrelationships in an IP network, since it fails to capture the complex interconnections of layer-2 network elements (switches, bridges, and hubs) that comprise each ethernet Local Area Network (LAN).

The complexity of performing ethernet topology discovery arises from the inherent transparency of ethernet bridge hardware. Endpoints are unaware of the presence of bridges in the network. The bridges themselves only communicate with their neighbors in the limited exchanges of the spanning tree protocol, and that is not used in all environments. The only useable MIB information maintained by switches and bridges is in the Address Forwarding Table (AFT) - the set of Media Access Control (MAC) addresses that are reachable from a port of a given node. If AFTs are complete, (that is, they contain all and only nodes that can be reached from a node's port). However, it is unrealistic to expect that the information in AFTs is complete for the real network environment.

In this paper we defines three constraints of AFTs, and proposes a tree chopping algorithm based on AFTs satisfying down constraint, which can discover the physical topology of a subnet accurately. The proposed algorithm decreases the demand for AFTs integrity dramatically, and is the loosest constraint for discovering physical topology which only relies on AFTs of down ports. i.e., if the down ports' AFTs of all nodes can define the network physical topology, then the topology can be discovered by the proposed tree chopping algorithm.

The paper is organized as follows. It starts with an overview of related work in section 2. Section 3 defines three AFT constrains and provides the detailed description of the proposed tree chopping algorithm. It also illustrates the application of our algorithm to one set of AFTs. Section 4 describes the simulation results and Section 5 concludes the work.

## 2. Related Work

The main reason of the hardness of physical topology discovery is:

1. Most of the current network topology tools collect and manage networks at the IP layer and require network managers to maintain layer-2 connections manually, furthermore, the layer-2 node's MIB does not provide information on its immediate neighbors.
2. Port's AFTs provide information of reachable nodes, but the completeness of AFTs cannot be guaranteed.

To overcome this difficulty, the IETF has proposed a "physical topology" MIB [1], but the proposal merely

reserves a portion of the MIB space without defining any protocol or algorithm for obtaining the topology information. Further, the IEEE 802.1AB Committee finished a proposal on a new layer-2 discovery protocol, called link layer discovery protocol. This allows layer-2 neighbors to notify one another of their presence. However, even if vendors would embrace the protocol, there is a large portion of legacy hardware in networks that still needs a general layer-2 network discovery protocol. A number of vendors have developed proprietary tools and protocols for inferring layer-2 connectivity between different network elements. Examples of such systems include Cisco's discovery protocol and extreme networks' extreme discovery protocol. Such tools, however, are typically based on vendor-specific extensions to MIBs and are not useful on a heterogeneous network comprising elements from multiple vendors.

Besides using neighbor discovery protocol, the physical topology discovery methods mainly can be classified into four ways:

1. Port traffic feature based.
2. Spanning Tree Protocol (STP) based.
3. Probing packets based, 4) AFT based.

The work [9, 17, 20] proposed to discover layer-2 topology using port traffic features. Briefly, their approach is based on trying to statistically map (i.e., correlate) the traffic patterns observed at the ports of different elements in the underlying network, and probabilistically inferring connections for ports with similar traffic characteristics. Their approach relies on statistical correlation, so it can only infer element connections with some (high) probability; furthermore, it is not at all clear if or how their proposed method would work in the presence of interconnections between network elements belonging to different IP subnets.

The work [21] proposed a STP based method to discover the connection relationship between physical ports. The basic idea of this method is to build connectivity between bridges using spanning tree information. It can discover all links (not only the active links) between bridges, but it fails to discover the path between bridges and hosts, besides the support of STP for all bridges limits its use.

Black *et al*. [2] listed some problems with finding a layer-2 topology using bridge MIB data. They proposed a new probing-packets-based method to find a layer-2 topology without querying network MIB information. However, their approach requires placing custom designed network daemons on each host in the network, which some network managers might find objectionable.

Overall, the above methods have some limitations for discovering layer-2 topology. AFT based method caused more and more attention recently. The initial algorithm developed by Breitbart *et al*. [3] depended on complete AFT data collected from every single element in the network. Breitbart *et al*. [4] also observed that for multi-subnet networks the network topology may not be unique even for the set of complete AFTs obtained from a simple Ethernet network. Bejerano *et al*. [5] proposed the first formal algorithm to discover the topology in presence of uncooperative elements (i.e., hubs.). The algorithm was too complex to understand and implement in practice. Furthermore, this method may not discover any topology if the given input set of AFTs defines a non-unique topology.

Zheng and Zhang [23] proposed a method only rely on AFTs of down ports of bridges. The method can build connectivity of tree nodes if the down ports' AFTs are complete. Lowekamp *et al*. [18] relaxed the dependency on complete AFTs information and proposed a necessary and sufficient condition for two AFTs to be connected ethernet topology (directly or indirectly). Their work also addressed the topologies that may contain uncooperative nodes. However, their approach can only be used in a single subnet.

Yantao *et al*. [22] proposed an algorithm based on "connections reasoning technique" that was claimed to be necessary and sufficient to discover the layer-2 topology even when the information provided by nodes MIBs is incomplete. However, their claim was not supported by proofs. Bejerano [6, 7] proposed a very simple layer-2 topology discovery method for multi-subnet networks. While his method discovers layer-2 network topology in a wide variety of cases, it cannot guarantee topology discovery. His method also requires completeness of input AFTs.

Gobjuka and Breitbart [8, 11, 12, 13, 14, 15] described the first formal method to determine whether a given set of complete AFTs define a unique topology when the network doesn't contain hubs. Their methods discover all network topologies when the MIB information defines more than one topology. Furthermore, they proposed criteria to decide the uniqueness of network topology from a complete set of AFTs when the network contains hubs. They proved that finding a layer-2 network topology for a given set of incomplete AFTs is a NP-hard problem even for single subnet networks and deciding whether a given set of AFTs defines a unique network topology is a co-NP-hard problem. They also described methods for inferring complete AFTs from incomplete information. This approach is used in heuristic that discovers the topology from incomplete AFTs.

## 3. Down Constraint Algorithm

### 3.1. Defining the Constraints

An Ethernet is a graph with two kinds of nodes: hosts, with a single link, and network elements, with multiple links. Ethernet requires that all redundant links have been eliminated (either through STP, or by wiring

rules); the graph is therefore a tree, with a single path between any two nodes. Each host is characterized by its distinct MAC address, thus a computer with multiple network interfaces is treated as multiple hosts. All devices and connections in the administrative domain form a tree which we specify topology-tree in this paper. We specify the management station R as the root of the topology-tree. We call the ports of switches engaged in the topology-tree as the active ports. The active port of a switch from which the switch communicate with the root R through the shortest path is called the up port, and the other active ports of the switch are called as the down ports. Without specifying, the ports mentioned in the later content are all the active ports.

If the node $S_i$ connects to the root $R$ passing the node $S_j$, we call $S_i$ is the descendant node of $S_j$ and $S_j$ is the ancestor node of $S_i$. $R$ is the ancestor of all nodes in the topology-tree. The relationship between $S_j$ and $S_i$ is called as the lineal connection. $S_j$ and $S_i$ are the lineal nodes of each other. We call the relationship that two nodes directly connect with each other as a parent-child relationship which is a specific case of the lineal connection. If the relationship between two nodes is not a lineal connection, we call this relationship as a collateral connection. Suppose $S_j$ connects (directly or indirectly) to the designated port $p$ of $S_i$, we say the port $p$ is the ancestor port of $S_j$ (and the ports of $S_j$), and say $S_j$ is the descendant node of the port $p$ and all ports of $S_j$ is the descendant ports of the port $p$.

Thus, we model the network as an undirected topology-tree $G =<V; E >$, where $V$ is a set of all network nodes and each element of $E$ represents a physical connection between two nodes. The network nodes contains bridges (network elements can provide their AFT mibs), hosts (computers and routers, node with multiple network interfaces is treated as multiple hosts), dumb devices (hubs and any network devices which their mibs cannot be accessed). Hence, $V=S+H+D$, $S$ denotes switches or bridges, $H$ denotes hosts, and $D$ denotes dumb devices.

Each port of a switch maintains an AFT. We denote the jth interface of a switch $S_i$ by $S_{ij}$. For each interface, the set of addresses that have been learned (by backward learning) on that interface is referred to as the AFT corresponding to $S_{ij}$ and is denoted by $A_{ij}$. Therefore, $A_{ij}$ is the set of MAC addresses that have been seen as source addresses on packet frames received at $S_{ij}$. We say that $A_{ij}$ is complete if $A_{ij}$ contains the MAC addresses of all network nodes from which frames can be received at $S_{ij}$.

We should notice the topology-tree difference between the network without a dumb device and the network with dumb devices. Each down port of the former has only one child, while each down port of the latter may have multiple children. Hence, normally a dumb device can be inferred by the number of the child of a down port. In our following discussion, we ignore dumb devices with only two edges (for example, the repeater), which cannot be found by any AFT based method and also other proposed topology discovering methods.

If the switched domain comprises only one subnet, then $A_{ij}$ corresponds to the set of nodes in $N$ that are reachable from $S_i$ via the interface $S_{ij}$ by a path in the switched domain spanning tree. In the case of multiple subnets, however, the above is not necessarily true [3, 4]. Hence, the complete AFT of a single subnet can define the unique topology of a single subnet, but it is not true for a multi-subnet [3, 4]. The topology-tree of a single subnet has the following properties.

- *Property* 1: The complete AFTs of a single subnet define a unique topology of the network.

*Proof*: For any node $S_i$ in the topology-tree $G$, if its $A_{ij}$ is complete, then $A_{ij}$ corresponds to the set of nodes in $V$ that are reachable from $S_i$ via the interface $S_{ij}$ by a path in the switched domain spanning tree. For any other node port $S_{mn}$ and the corresponding $A_{mn}$, there are two situations: 1)$A_{mn}\neq A_{ij}$, in this situation the connection of port $S_{ij}$ and $S_{mn}$ must be different, or it will lead to $A_{mn}=A_{ij}$; 2) $A_{mn}=A_{ij}$, in this situation there must exist a dumb device directly connecting the two ports. Hence, for either condition, we can define the unique connection, thus define the unique topology.

- *Property* 2: Ping all switches from R and if R belongs to $A_{ij}$, $S_{ij}$ is the up port of switch $S_i$.
- *Property* 3: The up port of $S_i$ is unique.
- *Property* 4: For a down port $S_{ij}$, if $S_x$ belongs to $A_{ij}$, then $S_x$ is the descendant node of $S_i$.

Property 3, 4, and 5 is the obvious tree property. Besides the above four properties, any other tree property can be used in topology tree, for example the transitivity of the nodes ancestor-descendant relationship. Since a complete AFT can define a unique topology of a subnet, we define three kinds of constraints.

- *Definition* 1: If the AFTs of a single subnet can define a unique topology, we call such AFTs satisfy minimal constraint of a single subnet.
- *Definition* 2: Suppose the root switch $R$ is designated, the up port AFT of each switch contains at least one ancestor switch's address (not null), if down ports' AFTs of the single subnet can define a unique topology, we call such down ports' AFTs satisfy Minimal Down Constraint (MDC) of a single subnet.
- *Definition* 3: Suppose the root switch $R$ is designated, the down port AFT of each switch contains at least one descendant's address (not null), if up ports' AFTs of the single subnet can define a unique right topology, we call such up ports' AFTs satisfy minimal up constraint of a single subnet.

Obviously, the complete AFTs satisfy minimal constraint of a single subnet, but AFTs satisfying minimal constraint is not necessarily complete. From the above definition we can see that the minimal constraint is the loosest constraint for constructing a unique topology based on AFTs. The MDC is the loosest down constraint for constructing a unique topology based on down ports AFTs. The minimal up constraint is the loosest up constraint for constructing a unique topology based on up ports AFTs.

Although the minimal constraint is the loosest constraint for a unique topology, the method how to get the topology by minimal constraint is obviously the hardest one. Methods discovering the topology based on minimal down/up constraint are comparatively easy. In this work, we focus on topology discovery based on AFTs satisfying MDC. We will study others in our future works.

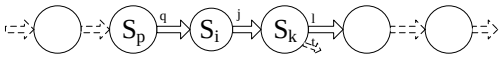


Figure 1. A path in a topology-tree.

- *Lemma* 1: For the AFTs satisfying MDC, the down port AFT of each node either contains its child's address or contains the child's reachable addresses at least from the child two down ports.

*Proof*: Figure 1 illustrates any path in a topology-tree. $S_i$ and $S_k$ denote any adjacent nodes, and $S_i$ is the father of $S_k$. We denote $B_{kt}$ as reachable addresses from down port $S_{kt}$. The AFTs satisfying MDC can define the topology based on down port AFT. Lemma 1 needs: 1) $S_k \in A_{ij}$, or 2) *if* $S_k \notin A_{ij}$, then $\exists a \in B_{kl} \wedge \exists b \in B_{kt}$, $\partial \{a, b\} \subseteq A_{ij}$. If the two conditions cannot be met, then it must be: 1) $A_{ij}$ is null; or 2) only exist one port, for example $S_{kl}$, $\partial (\exists a \in B_{kl} \wedge a \in A_{ij})$. In either situation, the node $S_i$ and $S_k$ can be exchanged in Figure 1, which does not influence the down port AFTs of any node in topology-tree. That is, the down port AFTs cannot define a unique topology, which contradicts with the MDC.

From lemma 1 we can know that if the child of a node only has one down port, the node down port must contain the address of its child. From lemma 1 we can see that the undirected topology-tree turns into a downward directed tree in the view of AFTs. A down port of each node has edges toward its descendants, one of the edges goes toward its child or at least two edges go toward its descendants from two down ports of its child. For such a tree, we put forward a very easy and efficient tree chopping algorithm to discover the original tree topology. The time complexity of the algorithm is $O(n^2)$, and $n$ is the switch number of the tree. The basic idea of our proposed algorithm is based on such a simple principle "if a branch and its leaves are cut off, and the cut branch in the tree turns into a new leaf, then the tree can be cut over". Before starting the algorithm, we need to prove some lemmas of AFTs satisfying MDC to support the proposed algorithm.

- *Definition* 4: For a switch, if all its down ports cannot reach any other switches, we call such a switch leaf switch.
- *Definition* 5: Except root switch, we call any non-leaf switch media switch.
- *Definition* 6: For a switch down port, if its child is a leaf switch, we call such a port leaf port**.**
- *Definition* 7: We call all hosts as leaves.
- *Lemma* 2: For the leaf switch, there is only one port whose AFT contains other switches' address.

*Proof*: According to definition 4, the down ports of a leaf switch cannot reach any switches, thus all the down ports AFT cannot contain any switch address. According to definition 2 and property 3, the up port AFT of each switch contains at least one ancestor switch's address, and the up port of each switch is unique. Hence, we get that for the leaf switch, there is only one port whose AFT contains other switches' address.

- *Lemma* 3: For the media switch, either at least two ports AFTs contain other switch address, or only one port AFT contains other switch address while at least one port exists whose AFT's addresses distribute in different ports AFTs of other switches.

*Proof*: According to definition 2 and property 3, the up port AFT of the media switch must contains an ancestor switch address. Moreover, it must have a down port can reach its child switch, or it will be a leaf switch. According to lemma 1, the down port AFT of the media switch either contains its child switch address or contains the child's reachable addresses at least from the child two down ports. When the down port AFT contains its child switch address or contains a descendant switch address, the media switch have at least two ports AFTs contain other switch address. When the down port AFT contains the child's reachable addresses which all belong to hosts, the host addresses must distribute in different ports AFT of other switches since they are reachable at least from the media switch's child two down ports.

- *Lemma* 4: If the leaf port AFT contains a switch address, the address must belong to its child leaf switch.

*Proof*: Lemma 4 is obvious according to definition 4 and 6.

## 3.2. Tree Chopping Algorithm

Based on the definitions and lemmas in section 3.1, we propose our tree chopping algorithm for AFTs satisfying MDC as follows:

1. Setting up the switch set $H$ (including the switch address and ports AFT).
2. Determining all leaf switch $S_k$: For all switches, select those switches whose only one port AFT (for example $A_{ij}$) contains other switch address as the

primary leaf switch set (lemma 2). Then get rid of mediate switches in the primary set based on lemma 3, we can get the final leaf switch set.

3. Deleting all leaf switches in *H*: Get rid of all leaf switches from *H*. If a down port AFT of a leaf switch has multiple leaves, the down port connects those leaves with a dumb device.
4. Updating AFTs in *H*: Change those addresses in $A_{ij}$ which are contained by the down ports of these cut leaf switches, and merge the repetitive items. This step turns all cut leaf switches into new leafs.
5. Goto 2, until only the root is left in *H*.

We use a case to illustrate the tree chopping algorithm. The topology-tree is shown in Figure 2-A hub connects host h8 and h9. The given AFTs are very incomplete. The topology cannot be discovered by complete AFTs based methods [3, 5] and down port complete AFTs based methods [10, 23], and even cannot be discovered by some of those incomplete AFTs based methods with higher time complexity [6, 7, 8, 11, 12, 13, 14, 15, 17, 22].

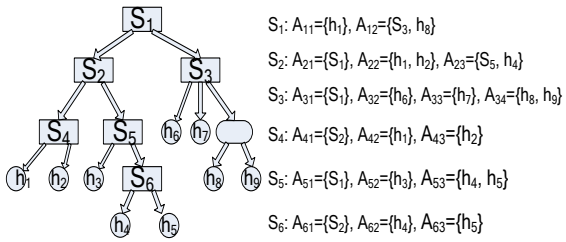


Figure 2. A Classic topology-tree and its AFTs.

We use our propose tree chopping algorithm to discover the topology. When the root $S_1$ is specified (usually we choose the switch connects with the router directly in a subnet as root in real network), put all switches addresses and AFTs into *H*. Since $S_1$ is the root, we first select the primary leaf switch set based on lemma 2 from the remainders. From Figure 2 we can see that two ports (port *1* and *2*) of $S_2$ contain switch address, the others only have one port containing switch address. Thus, the primary leaf switch set is: $\{S_3, S_4, S_5, S_6\}$.

Further, for these ports with multiple leaves such as $A_{34}$ of $S_3$ and $A_{53}$ of $S_5$, the host $h_8$ and $h_9$ in $A_{34}$ do not distribute in other switches different ports, but the host $h_4$ and $h_5$ in $A_{53}$ distribute in $A_{62}$ and $A_{63}$. Based on lemma 3, $S_5$ is cut out from the leaf switch set. Thus, the final leaf switch set is: $\{S_3, S_4, S_6\}$. We can judge that $S_{53}$ connects with leaf nodes $h_8$ and $h_9$ with a dumb device. We cut all the leaf switches and their leaves, and turn these cut leaf switches into new leaves, also update the AFTs in *H*. The new tree and new H is shown in Figure 3.

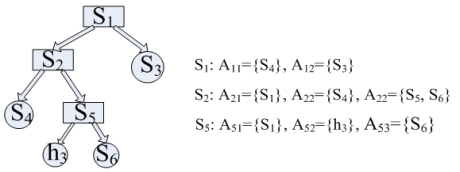


Figure 3. The Tree after the first cut and H.

After the first cut, the remaining switches set is: $\{S_1, S_2, S_5\}$. Other switches turn into leaf nodes. All ports AFTs are updated. Based on lemma 2 and lemma 3, we select $\{S_5\}$ as the new leaf switch set, and continue the cutting process until the root $S_1$ is left. The remaining cutting process is illustrated in Figure 4.

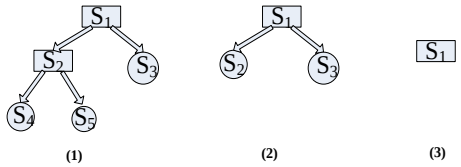


Figure 4. The Remaining cutting process.

The cutting nodes each time record the direct connections, thus the topology is discovered. AFTs satisfying MDC can be very useful for the scenario that the manager of root switch downloads AFTs of the subnet [8, 9, 10, 11, 12, 13, 14, 15]. Obviously, MDC is much looser than the complete AFTs of down ports. From definition 2 we know that MDC is the loosest constraint defining a unique topology based on AFTs of down ports. Down ports AFTs do not meet MDC cannot define a unique topology.

Our algorithm is proposed strictly based on lemma 1, 2, 3, and 4. Hence, a unique topology can be discovered by our proposed algorithm if the AFTs satisfy MDC. Our proposed algorithm is very simple and easy with a time complexity of $O(n^2)$, *n* indicates the switch nodes number. Besides, our proposed algorithm only needs AFTs of switches. This is very practical for real network since most hosts do not configure SNMP for privacy.

## 4. Evaluation

We use BRITE [13] and NS2 to evaluate our algorithm. We used BRITE [13] to generate Power-Law-based random tree-like networks with different number of nodes that vary from 50 to 1000. Through revising BRITE code, we can directly export the nodes connections generated by BRITE to NS2 for further analysis. Figure 5 is an example topology with 60 nodes generated by BRITE and shown by NS2.

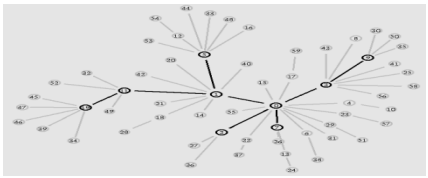

Figure 5. The Topology Generated by BRTIE and Shown in NS2.

We use BA model when generating nodes connections by BRTIE. BA model suggests two possible causes for the emergence of a power law in the frequency of out degrees in network topologies: incremental growth and preferential connectivity. Incremental growth refers to growing networks that are formed by the continual addition of new nodes, and thus the gradual increase in the size of the network. Preferential connectivity refers to the tendency of a new node to connect to existing nodes that are highly connected or popular. We choose the incremental growth mode and the first node 0 generated as the root.

The network traffic is generated in such a way: The root node 0 sends a FTP packet to other nodes one by one each second. That is, node 0 send a packet to node 1 in the first second, send a packet to node 2 in the next second, etc., Since FTP uses TCP in transport layer, which leads to an "ack" for every packet. The AFT can be constructed using backward leaning based on the traffic. AFTs satisfying MDC can be got by AFTs aging and artificial deletion.

The nodes generated by BRITE can be classified into three types: switches, hosts and dumb devices. We normally classify all leaf nodes with degree 1 as hosts, such as the node 50; we classify a small number of nodes with degree greater than 2 as dumb devices, such as node 2 and 9; other nodes are classified as switches, such as node 3 and 1. We should notice that nodes with degree 2 cannot be classified as dumb devices, such as node 6 and 18, because such devices cannot be discovered by any AFT-based methods. After classification, about 16% of these nodes were switches and the rest were hosts and dumb devices.

Our experiments were run on a Pentium 4 2.4 GHZ computer with 2G of RAM and Windows XP operating system. For all the topology generated by BRITE, as long as the constructed AFTs satisfying the MDC, our proposed algorithm is able to correctly determine the topology, which further testifies that a unique topology can be discovered by our proposed algorithm if the AFTs satisfy MDC.

Although correctness of the topology is the most important criterion when judging the performance of this algorithm, its time performance may also be important in some applications. Figure 6 shows the time consumed when calculate different nodes topology. The figure is illustrated using a double Y-axis. The X-axis indicates different nodes number of different topologies generated. The Y-axis 1 indicates time used by our algorithm when discovering different nodes topology, which marked with '□'. The Y-axis 2 indicates the switches number corresponding to different nodes topology, which is marked with 'x'. From Figure 6, we can see that the topology calculating by our algorithm is very fast for a given set of AFTs. Hence, the majority of the execution time is spent downloading the AFTs from the switches in real network.

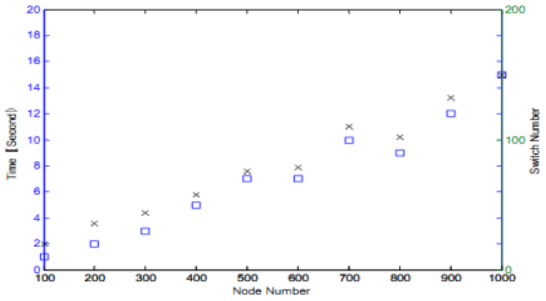


Figure 6. Time to calculate the topology.

Beside, from Figure 6 we can see that the time calculating topology increases with the number of network nodes. However, some exceptions exist, for example, time consuming in discovering 500 nodes topology is almost same as time in 600 nodes discovery, time spending in 800 nodes topology discovery is still less than that of 700 nodes discovery. The reason is that switches number is almost same for 500 and 600 nodes topology, and switches number of 800 nodes is less than 700 nodes. From Figure 6 we can see that the time coordinates with the switch number. This result further explains that our proposed algorithm is solely based the AFTs of switches.

The evaluation of our time performance can be further verified by our tree chopping algorithm. From the algorithm we can see that all nodes need to be chopped from leafs to root, the time consumed of the process is $O(n)$, which n is the switch number. However The execution of step 2 and step 4 need to search all switches AFT, thus the time consumed in step 2 and step 4 is also $O(n)$. That is, chopping one node need $O(n)$, and chopping all nodes need $O(n^2)$, thus the time complexity is $O(n^2)$ of our method.

At present the topology discovery methods based on down ports AFTs [10, 23] all demands the completeness of down ports AFTs. The topology cannot be discovered correctly using the methods [10, 23] if down ports AFTs are incomplete. Lowekamp *et al*. [18] relaxed the dependency on complete AFTs information, but their method can only be applied in a subnet. Other work [6, 7, 8, 11, 12, 13, 14, 15, 17, 22] also proposed methods to discover the topology in a network with incomplete AFT, but the discovery process is complex and the calculation time is basically $O(n^3)$ in these method. The concrete comparison of our proposed method with other methods is listed in Table 1.

Table 1. Comparison of AFT topological methods.

| Methods | Completeness of AFT | Network | Time Complexity |
|---|---|---|---|
| **[10,23]** | Completeness of down ports | Subnet | $O(n^2)$ |
| **[18]** | Not complete | Subnet | $O(n^2)$ |
| **[6, 7, 8,11, 12, 13, 14, 15,17,22]** | Not complete | Network | $O(n^3)$ |
| **Our method** | Not complete | Network | $O(n^2)$ |

From Table 1 we can see that our method gains some advantages compared with the existing methods. Our method is very easy only needs AFTs of switches.

This is very practical for real network. Our next work will apply the tree chopping algorithm in real network condition to further prove the efficiency and applicability of our algorithm in real network.

## 5. Conclusions

In this work, we propose an algorithm which can discover a unique subnet topology based on incomplete AFTs satisfying down constraint. Our proposed algorithm decreases the demand of AFTs integrity dramatically, and can discover a unique topology based on down ports AFTs meeting MDC. Our proposed algorithm can also be used in the switch domain of multiple subnets as long as AFTs of the topology-tree meets lemma 1, and can discover a unique topology of the multiple-subnets switch domain if lemma 1 is met.

Since the AFTs of a switch domain of multiple subnets may not define a unique topology. There are cases which lemma 1 is defied. In our future work, we will focus on topology discover in such cases. At the same time, we will engage in the research on topology discovery based on incomplete AFTs satisfying minimal up constraint and minimal constraint, which also provides an open study for other researchers. Topology discovery in mobile networks [16] is also an urgent and important study in our future research.

## Acknowledgments

This work was supported by the National Natural Science Foundation of China under Grant No. 61462009, the China Postdoctoral Science Foundation under No. 2015M582832, the Jiangsu Post-doctor Research Fund of China under No. 1402138C, the Natural Science Foundation of Guangxi under Grant No. 2014GXNSFAA118358, the Jiangsu Future Network Research Project under No. BY2013095-1-15.

## References

[1] Bierman A. and Jones K., "Physical Topology MIB," *Internet RFC-2922*, 2000.

[2] Black R., Donnelly A., and Fournet C., "Ethernet Topology Discovery Without Network Assistance," *in Proceedings of the 12th IEEE International Conference on Network Protocols*, Berlin, pp. 328- 339, 2004.

[3] Breitbart Y., Garofalakis M., Martin C., Rastogi R., Seshadri S., and Silberschatz A., "Topology Discovery in Heterogeneous IP Networks," *in Proceedings of Computer Communications 19th Annual Joint Conference of the IEEE Computer and Communications Societies*, Tel Aviv, pp. 265-274, 2002.

[4] Breitbart Y., Garofalakis M., Jai B., Martin C., Rastogi R., and Silberschatz A., "Topology Discovery in Heterogeneous IP Networks: The NetInventory System," *IEEE/ACM Transactions on Networking*, vol. 12, no. 3, pp. 401-414, 2004.

[5] Bejerano Y., Breitbart Y., Garofalakis M., and Rastogi R., "Physical Topology Discovery for Large Multi-Subnet Networks," *in Proceedings of 22nd Annual Joint Conference of the IEEE Computer and Communications Societies*, SanFrancisco, pp. 342-352, 2003.

[6] Bejerano Y., "Taking the Skeletons Out of the Closets: A Simple and Efficient Topology Discovery Scheme for Large Ethernet Lans," *in Proceedings of 25th IEEE International Conference on Computer Communications*, Barcelona, pp. 1-13, 2006.

[7] Bejerano Y., "Taking the Skeletons out of the Closets: A simple and Efficient Topology Discovery Scheme for Large Ethernet Lans," *IEEE/ACM Transactions on Networking*, vol. 17, no. 5, pp. 1385-1389, 2009.

[8] Breitbart Y. and Gobjuka H., "Characterization of Layer-2 Unique Topologies," *Information Processing Letters*, vol. 105, no. 2, pp. 52-57, 2008.

[9] Dawes N., Schenkel D., and Slavitch M., "Method of Determining the Topology of a Network of Objects," *U.S. Patent 6,231,997*, 2002.

[10] Fu C., Jiahai Y., and Yang Y., "New Algorithms on IP Network Topology Discovery and Its Implement," *Acta Electronica Sinica*, vol. 36, no. 8, pp. 1620-1625, 2008.

[11] Gobjuka H. and Breitbart Y., "Characterization of Layer-2 Unique Topologies in Multisubnet Local Networks," *in Proceedings of Conference on Local Computer Networks*, Tampa, pp. 540-542, 2006.

[12] Gobjuka H. and Breitbart Y., "Ethernet Topology Discovery for Networks with Incomplete Information," *in Proceedings of 16th International Conference on Computer Communications and Networks*, Honolulu, pp. 631-638, 2007.

[13] Gobjuka H. and Breitbart Y., "Finding Ethernet-Type Network Topology is not Easy," Technical Report, 2007.

[14] Gobjuka H. and Breitbart Y., "Discovering Network Topology of Large Multisubnet Ethernet Networks," *in Proceedings of 32nd IEEE Conference on Local Computer Networks*, Dublin, pp. 428-435, 2007.

[15] Gobjuka H. and Breitbart Y., "Ethernet Topology Discovery for Networks with Incomplete Information," *IEEE/ACM Transactions on Networking*, vol. 18, no. 4, pp. 1220-1233, 2010.

[16] Kef M., Chergui L., and Benmohammed M., "Self-organization and Topology's Control for Mobile Ad-hoc Networks," *The International Arab Journal of Information Technology*, vol. 8,

no. 3, pp. 227-234, 2011.
[17] Lin Q., Jianzhong Z., and Gongyi W., "Reaearch of Physical Network Topology Discovery at Layer 2 Based on Octets," *Computer Engineering and Application*, vol. 22, pp. 171-172, 2002.
[18] Lowekamp B., O'Hallaron D., and Gross T., "Topology Discovery for Large Ethernet Networks," *in Proceedings of Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, San Diego, pp. 428-435, 2011.
[19] Siamwalla R., Sharma R., and Keshav S., "Discovering Internet Topology," Techical Report, 1999.
[20] Schenkel D., Slavitch M., and Dawes N., "Method of Determining Topology of a Network of Objects Which Compares the Similarity of the Traffic Sequences/Volumes of a Pair of Devices," *U.S. Patent 5,98,462*, 1999.
[21] Son M., Joo B., Kim B., and Lee J., "Physical Topology Discovery for Metro Ethernet Networks," *Electronics and Telecommunications Research Institute Journal*, vol. 27, no. 4, pp. 355-366, 2005.
[22] Yantao S., Zimei W., and Ziqiang S., "A Method of Topology Discovery for Switched Ethernet Based on Address Forwarding Tables," *Journal of Software*, vol. 17, no. 12, pp. 2565-2576, 2006.
[23] Zheng H. and Zhang G., "An algorithm for physical Network Topology Discovery," *Journal of Computer Research and Development*, vol. 39, no. 3, pp. 264-268, 2002.

**Bin Zhang** received his Ph.D. degree in Department of Computer Science and Technology, Tsinghua University, China in 2012. He received the Bachelor degree in computer software from Zhengzhou University, China in 1998 and the Master's degree in network information security from Shanghai Jiaotong University, China, in 2005. During his Ph.D. career, he publish more than 20 papers in refereed international conferences (NOMS, IM, LCN, IWQoS, APNOMS, etc) and journals (the Computer Journal, JCST, Journal of Software). He is a post doctor in Nanjing Telecommunication Technology Institute，Nanjing, China. His current research interests focus on Internet architecture and its protocols, IP routing technology, network measurement, network management, etc.,

**Xingchun Diao** is the headmaster and also a professor in Nanjing Telecommunication Technology Institute，Nanjing, China. He has published more than 100 articles in refereed international conferences and journals, and two books on data quality and management. Xinchun's research interests include big data processing, data quality, network measurement, network management, etc., He also serves as a TPC member for several international conferences.

**Donghong Qin** received his Ph.D. degree in Department of Computer Science and Technology, Tsinghua University, China in 2013. He publish more than 20 papers in refereed international conferences and journals. He is a professor in School of Information Science and Engineering, GuangXi University for Nationalities, Nanning, China. His current research interests focus on Internet architecture and its protocols, IP routing technology, network management, etc.,

**Yi Liu** received his Ph.D. degree in Institute of Command and Control Engineering, Army Engineering University of PLA, China in 2018. He received the Bachelor degree in network engineering from Xi'an University of Posts & Telecommunications, China in 2011 and the Master's degree in computer applications from PLA University of Science and Technology, China, in 2014. He has published about 15 papers in refereed international conferences (ICIQ, ICRIS, BIC-TA, CCF Bigdata etc.) and journals (MPE, Systems Engineering and Electronics, Journal of Software etc.). He is now a assistant professor in National Innovation Institute of Defense Technology, Beijing, China. His current research interests include Evolutionary algorithms, Data quality, Robot operating system, computer network, etc.

**Yun Yu** received the Bachelor degree in College of Telecommunications & Information Engineering from Nanjing University of Posts and Telecommunications, China in 2003 and the Master's degree in Software Engineering from University Of Electronic Science And Technology Of China, in 2010. He publish more than 10 papers in refereed international conferences (ICEMC, etc) and journals (High Performance Computing Technology, Journal of Modern Militart Communications, etc). He is now an engineer in The Sixty-third Research Institute,National University of Defense Technology, Nanjing, China. His current research interests focus on networking protocol, IP routing technology, network measurement, etc.