

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

JUNIPER NETWORKS, INC.,
Petitioner,

v.

PORTSMOUTH NETWORK CORPORATION,
Patent Owner.

Case No. IPR2024-00893
Patent No. 6,917,986

DECLARATION OF BRUCE MCNAIR

Juniper Exhibit 1003 Juniper v Portsmouth Network IPR2024-00893

TABLE OF CONTENTS

I.	INTRODUCTION	1
A.	Engagement	1
B.	Background and Qualifications	2
1.	Educational Background	2
2.	Professional Experience	2
C.	Bases of My Opinions and Materials Considered.....	5
D.	My Understanding of Patent Law	5
1.	Anticipation	7
2.	Obviousness.....	7
II.	DESCRIPTION OF THE RELEVANT FIELD AND THE RELEVANT TIME FRAME	11
III.	BACKGROUND	11
A.	Networking fundamentals	11
1.	Graph theory	11
2.	Spanning tree	15
3.	Network protocols	16
B.	Setting up a Spanning Tree Network.....	20
1.	STP Bridge Roles	27
2.	STP Bridge Port Roles	34
3.	STP Bridge Port States.....	42
4.	Maintaining STP Using Bridge Databases	44
C.	STP Self-Healing Process	47
1.	Link Failure Detection	50

2. Link Failure Recovery	52
D. Extensions to STP.....	53
1. IEEE 802.1W-2001 Rapid Spanning Tree Protocol.....	54
2. IEEE 1999	58
IV. THE '986 PATENT	60
A. The Challenged Claims.....	63
B. Prosecution History.....	64
C. Level of Ordinary Skill in the Art	64
V. UNPATENTABILITY GROUNDS.....	65
A. Ground 1: Claims 1-3, 6-15, and 18-24: Obvious Over Gai in View of IEEE 1999	65
1. U.S. Patent No. 6,032,194 (“Gai”)	65
a. Port Functions And Switch Connectivity	65
b. Port Reconfiguration After Link Failure	67
c. Dummy Traffic	71
d. Network Initialization Commands	72
2. IEEE 1999	75
3. Gai in View of IEEE 1999	78
4. Claim 1	83
a. [1PRE] A method for network communication, comprising:.....	83
b. [1A] coupling an edge switch in a network to one or more access switches via at least first and second redundant links, respectively connected to first and second ports of the edge switch;.....	83

c.	[1B] placing the first port in an active state, while placing the second port in a blocking state;	87
d.	[1C] conveying communication traffic over the first link while the first port is in the active state, thereby causing each of the switches to build a respective database for use in forwarding the traffic;	88
e.	[1D] responsive to a failure associated with the first link, placing the second port in the active state and the first port in the blocking state; and	90
f.	[1E] sending dummy traffic from the edge switch over the second link to the one or more access switches, so as to cause each of the one or more access switches to modify its respective database responsive to the second port being in the active state.	94
5.	Claim 2: A method according to claim 1, wherein placing the second port in the blocking state comprises configuring the first and second ports so that no loop is created in the network between the edge switch and the one or more access switches.	100
6.	Claim 3: A method according to claim 2, wherein configuring the first and second ports comprises configuring the edge switch so as to decrease a radius of a spanning tree established in the network.	102
7.	6. Claim 6: A method according to claim 1, wherein sending the dummy traffic comprises sending data frames originating at the edge switch while having source addresses corresponding to network addresses of other entities in the network that are accessible to the access switches via the edge switch.....	103
8.	Claim 7: A method according to claim 6, wherein sending the data frames comprises copying the source addresses for the data frames from the database of the edge switch.	103
9.	Claim 8: A method according to claim 7, wherein	104
a.	[8A] the edge switch has three or more ports, including the first and second ports, and.....	104

b. [8B] wherein prior to the failure, the database comprises entries associated with a plurality of the ports, and	105
c. [8C] wherein copying the source addresses comprises using the network addresses in all the entries that are not associated with the first port as the source addresses of the data frames.....	107
10. Claim 9: A method according to claim 6, wherein sending the data frames comprises incorporating a Virtual Local Area Network (VLAN) tag in a header of the data frames.	108
11. Claim 10: A method according to claim 6, wherein sending the data frames comprises multicasting the data frames to all of the one or more access switches.....	109
12. Claim 11: A method according to claim 1, wherein sending the dummy traffic comprises sending data frames downstream from the edge switch toward the access switches.....	109
13. Claim 12	110
a. [12A] A method for network communication, comprising: coupling a first bridge in a network to a second bridge located downstream of the first bridge, by connecting at least first and second redundant links to respective first and second ports of the first bridge so as to communicate with the second bridge;	110
b. [12B] placing the first port in an active state, while placing the second port in a blocking state;	110
c. [12C] conveying communication traffic over the first link while the first port is in the active state, thereby causing the bridges to build respective databases for use in forwarding the traffic;	111
d. [12D] responsive to a failure associated with the first link, placing the second port in the active state and the first port in the blocking state; and	111
e. [12E] sending dummy traffic from the first bridge over the second link downstream to the second bridge,	

so as to cause the second bridge to modify its database responsive to the second port being in the active state.....	111
14. Claim 13	112
a. [13A] A network switching device, comprising: a network port, adapted to be coupled over a network trunk link to a core switch in a network so as to exchange communication traffic with the core switch;.....	112
b. [13B] first and second access ports, adapted to be coupled via first and second redundant network access links to one or more access switches serving network users; and	113
c. [13C] a protocol processor, adapted to place the first access port in an active state, while placing the second access port in a blocking state, so that the communication traffic is conveyed over the first link while the first port is in the active state thereby causing each of the access switches to build a respective access switch database for use in forwarding the traffic, the protocol processor being further adapted, responsive to a failure associated with the first link, to place the second access port in the active state and the first port in the blocking state and to send dummy traffic through the second access port over the second link to the one or more access switches, so as to cause each of the one or more access switches to modify its respective access switch database responsive to the second access port being in the active state.	113
15. Claim 14: A device according to claim 13, wherein the protocol processor is adapted to configure the first and second access ports so that no loop is created in the network between the edge switch and the one or more access switches.	114
16. Claim 15: A device according to claim 14, wherein the protocol processor is further adapted to configure the first	

and second access ports so as to decrease a radius of a spanning tree established in the network.	115
17. Claim 18: A device according to claim 13, wherein the dummy traffic comprises data frames originating at the edge switch while having source addresses corresponding to network addresses of other entities in the network that are accessible to the access switches via the edge switch.	115
18. Claim 19: A device according to claim 18, and	116
a. [19A] comprising a memory, ...	116
b. [19B] wherein the protocol processor is adapted to build an edge switch database in the memory responsive to the communication traffic prior to the failure associated with the first link, and to copy the source addresses for the data frames from the edge switch database.	116
19. Claim 20: A device according to claim 19, wherein prior to the failure, the edge switch database comprises entries associated with a plurality of the ports, and wherein the protocol processor is adapted to use the network addresses in all the entries that are not associated with the first access port as the source addresses of the data frames.	117
20. Claim 21: A device according to claim 18, wherein the protocol processor is adapted to incorporate a Virtual Local Area Network (VLAN) tag in a header of the data frames.	118
21. Claim 22: A device according to claim 18, wherein the protocol processor is adapted to multicast the data frames to all of the one or more access switches.	118
22. Claim 23: A device according to claim 13, wherein the dummy traffic comprises data frames sent downstream from the edge switch toward the access switches.	119
23. Claim 24	119
a. [24A] A bridge device for use in a communication network having a spanning tree root, the device	

comprising: a designated port, adapted to be coupled over an upstream link in the network so as to communicate with the root;.....	119
b. [24B] first and second downstream ports, adapted to be coupled via first and second redundant links to a further bridge device located downstream in the network relative to the spanning tree root; and.....	119
c. [24C] a protocol processor, which is adapted to place the first downstream port in an active state, while placing the second downstream port in a blocking state, so that communication traffic is conveyed over the first downstream link while the first port is in the active state, thereby causing the bridge devices to build respective databases for use in forwarding the traffic, the protocol processor being further adapted, responsive to a failure associated with the first link, to place the second downstream port in the active state and to send dummy traffic over the second link downstream to the further bridge device, so as to cause the further bridge device to modify its database responsive to the second downstream port being in the active state.	120
B. Ground 2: Claims 4, 5, 16, 17: Obvious over Gai in View of IEEE 1999, Further in View of Jain	121
1. U.S. Patent No. 6,330,229 (“Jain,” EX1006).....	121
2. Gai in View of IEEE 1999, Further in View of Jain	122
3. Claim 4: A method according to claim 1, and comprising, responsive to placing the second port in the active state, updating the database of the edge switch so that entries in the database point to the second port instead of the first port.	123
4. Claim 5: A method according to claim 4, wherein updating the database comprises changing the entries in the database to point to the second port before receiving the communication traffic from the one or more access switches on the second port.	124

5. Claim 16: A device according to claim 13, and.....	125
a. [16A] comprising a memory, ...	125
b. [16B] wherein the protocol processor is adapted to build an edge switch database in the memory responsive to the communication traffic prior to the failure associated with the first link.....	125
c. [16C] and, responsive to placing the second access port in the active state, to update the edge switch database so that entries in the edge switch database point to the second access port instead of the first access port.	125
6. Claim 17: A device according to claim 16, wherein the protocol processor is adapted to update the entries in the database to point to the second access port before receiving the communication traffic from the one or more access switches on the second por.....	126
VI. CONCLUSION	126

I, Bruce McNair, declare:

I. INTRODUCTION

A. Engagement

1. I have been retained on behalf of Juniper Networks, Inc. (“Juniper” or “Petitioner”) as an independent technical expert in the above-captioned IPR proceeding (the “IPR”). This document provides my opinions concerning the patentability of claims in U.S. Patent No. 6,917,986 (EX1001, “the ’986 patent”), specifically claims 1-24 (the “challenged claims”). I make this Declaration in support of Juniper’s IPR petition.

2. For my work as an expert in this matter, I am being compensated for my services at my standard rate of \$650/hour, plus actual expenses. My hourly compensation is based solely on the amount of time that I devote to activity related to this matter and is in no way contingent on the nature of my findings, the presentation of my findings in testimony, or the outcome of this or any other proceeding. Other than my hourly rate (and reimbursement for expenses) I have no personal or financial stake or interest in the outcome of this proceeding. I do not have any expectation or promise of additional business with the Petitioner or its counsel in exchange for my work on the matter or my opinions explained herein.

3. I make this Declaration based on my personal knowledge, including my education, training, research, and professional experience.

4. A true and correct copy of my Curriculum Vitae (“CV”) is provided as EX1004. This document provides further details about my background and experience.

B. Background and Qualifications

1. Educational Background

5. I received my Masters of Engineering (M.E.) degree in the field of Electrical Engineering from Stevens Institute of Technology (“Stevens”) in 1974 and my Bachelor of Engineering (B.E.) degree in the field of Electrical Engineering in 1971 from Stevens as well. I completed course work and passed the qualifiers for the PhD program in Computer Science at Stevens, as well.

2. Professional Experience

6. I am a Distinguished Service Professor of Electrical and Computer Engineering at Stevens Institute of Technology in Hoboken, NJ. I have studied and practiced in the fields of electrical engineering, computer engineering, and computer science for over 50 years, and have been a Professor of Electrical and Computer Engineering since 2002. I have taught undergraduate and graduate courses including wireless networks and security of communications networks. Since I retired from my full time faculty position at Stevens in 2017, I have continued to teach on-line versions of my graduate courses.

7. I am the Founder and Chief Technology Officer of Novidesic Communications, LLC, a technology consulting company. Prior to starting Novidesic and joining the faculty at Stevens in 2002, I spent 24 years at AT&T Bell Laboratories (“Bell Labs”). My most recent work for the last eight years at Bell Labs included research into next generation (4G and beyond) wireless data communications systems, including high-speed, high-mobility wide area networks as well as range and speed extensions to IEEE 802.11(a & b) wireless LANs. I also studied the interactions between IEEE 802.11 WiFi™ networks and IEEE 802.15.1 Bluetooth™ networks. For the previous seven years, I supervised the Bell Labs Security and System Reliability Architecture Group that served as subject matter experts in network reliability, security, authentication techniques, and encryption. Part of this work involved studying the security and network reliability of networks similar to the subject matter of the ‘986 patent. Before that, my activities included developing encryption hardware, secure voice architecture studies, high- speed voice-band modems, and public data network protocols. My first position at Bell Labs involved the design of a public data network using network-based functional servers interacting with client devices very similar to today’s Internet. Part of my work on that project involved studying potential failure and recover mechanisms for a distributed network.

8. Before joining Bell Labs, I spent seven years developing military communications systems for the U.S. Army Electronics Command and ITT Defense Communications Division. My responsibilities included developing cryptographic and ECCM techniques for portable radio systems, TEMPEST technology, and state-of-the-art speech compression techniques.

9. Previous to my work on military radio systems, I was at ITT Defense Communications Division developing secure satellite communications systems for the White House Communications Agency and the first hardware implementation of a Linear Predictive Speech coder for the National Security Agency and the Naval Research Laboratory.

10. Throughout my career in every one of the positions cited above, I have engaged in the design of numerous data networks with issues similar to those described in the '986 patent.

11. Two of the graduate courses I have taught at Stevens are directly related to the subject matter of the subject patent. I have taught CpE691-Information System Security several dozen times since I took over the course in 2002 and I have taught EE584-Wireless System Security almost sixty times since I developed the course in 2003. Both of these courses address IP networks and their failure modes.

12. I am a Life Senior Member of the IEEE and belong to the Communications and Signal Processing Societies. I have served as the Secretary of the IEEE Communications Society Communications Security Committee.

C. Bases of My Opinions and Materials Considered

13. In developing my opinions relating to the '986 patent, I have considered the materials cited herein. I also rely on my experience and education with respect to graph theory, data network operation and system reliability.

D. My Understanding of Patent Law

14. In developing my opinions, I discussed various relevant legal principles with Petitioner's attorneys. I have relied upon such legal principles, as explained to me, while forming the opinions set forth in this Declaration. My understanding in this respect is as follows:

15. I understand that *inter partes* review ("IPR") is a proceeding before the United States Patent & Trademark Office ("USPTO") for evaluating the patentability of an issued patent's claims based on prior-art patents and printed publications.

16. I understand that, in this proceeding, Juniper has the burden of proving that the challenged claims of the '986 patent are unpatentable by a preponderance of the evidence. I understand that a "preponderance of the evidence" means that the evidence establishes that a fact or conclusion is more likely true than not true.

17. I understand that patent claims can be independent or dependent. I understand that a dependent claim must reference a claim previously set forth, and then must specify a further limitation of the claimed subject matter. I also understand that a dependent claim is treated as incorporating by reference all limitations of the claim(s) from which it depends. I understand that an independent claim does not usually reference or incorporate limitations from other claims.

18. I understand that, in IPR proceedings, claim terms in a patent are given their ordinary and customary meaning as understood by a person of ordinary skill in the art ("POSA") in the context of the entire patent and the prosecution history pertaining to the patent. I discuss the qualifications of the POSA relevant to the '986 patent below. I understand that if the patent specification provides a special definition for a claim term that differs from the meaning the term would otherwise possess, the specification's special definition takes precedence. I have applied these standards in preparing the opinions in this Declaration.

19. I understand that, for an invention claimed in a patent to be patentable, it must be, among other things, new (and consequently not anticipated) and not obvious in view of the prior art. My understanding of these two legal standards is set forth below.

1. Anticipation

20. I understand that, for a patent claim to be “anticipated” by the prior art (and therefore not novel), each and every claim limitation of the claim must be disclosed, actually or inherently, in a single prior art reference. I understand that a claim limitation is disclosed for the purpose of anticipation if a POSA would have understood the reference to disclose the limitation based on inferences that a POSA would reasonably be expected to draw from the explicit teachings in the reference when read in light of the POSA’s knowledge and experience.

21. I understand that a claim limitation is inherent in a prior art reference if that limitation is necessarily present when practicing the teachings of the reference, regardless of whether a POSA recognized the presence of that limitation in the prior art.

2. Obviousness

22. I understand that a patent claim may be unpatentable if it would have been obvious to a POSA in view of a single prior art reference or a combination of prior art references.

23. I understand that a patent claim is obvious if the differences between the subject matter of the claim and the prior art are such that the claimed subject matter, as a whole, would have been obvious to a POSA at the time the invention was made. Specifically, I understand that the obviousness question involves a consideration of: the scope and content of the prior art; the differences between the prior art and the claims at issue; the knowledge of a POSA; and, if present, any “secondary considerations” of non-obviousness. At this stage I am unaware of the Patent Owner identifying any alleged secondary considerations of non-obviousness with respect to the '986 patent, but if any secondary considerations are later presented, I would be happy to evaluate and comment thereon if asked to do so.

24. I understand that for a claimed invention to be obvious, a POSA must have had a reason for combining teachings from multiple prior art references (or for altering a single prior art reference, in the case of obviousness in view of a single reference) in the fashion proposed.

25. I have also been informed that in considering whether a combination of elements recited in a patent claim would have been obvious, I may assess whether there are apparent reasons to combine known elements in the prior art in the manner claimed in view of interrelated teachings of multiple prior art references, the effects of demands known to the design community or present in the marketplace, and/or the background knowledge possessed by a POSA. I have

been informed that other principles may be relied upon in evaluating whether a claimed invention would have been obvious, and that these principles include the following:

- A combination of familiar elements according to known methods is likely to be obvious when it does no more than yield predictable results;
- When a device or technology is available in one field of endeavor, design incentives and other market forces can prompt variations of it, either in the same field or in a different one, so that if a POSA can implement a predictable variation, the variation is likely obvious;
- If a technique has been used to improve one device, and a POSA would recognize that it would improve similar devices in the same way, using the technique is obvious unless its actual application is beyond a POSA's skill;
- An explicit or implicit reason to modify a prior art reference or to combine two prior art references to arrive at the claimed invention may demonstrate obviousness, and this reason may come from a prior art reference or from the knowledge or common sense of a POSA;
- Market demand, rather than scientific literature, can drive design trends and may show obviousness;

- In determining whether the subject matter of a patent claim would have been obvious, neither the particular motivation nor the avowed purpose of the named inventor controls;
- One consideration supporting obviousness of a patent claim is if there existed at the time of invention a known problem for which there was an obvious solution encompassed by the claim;
- Any need or problem known in the field of endeavor at the time of invention can provide a reason for combining the elements in the manner claimed;
- Common sense teaches that familiar items may have obvious uses beyond their primary purposes, and in many cases a POSA will be able to fit the teachings of multiple patents together like pieces of a puzzle;
- A POSA is a person of ordinary creativity, and is not an automaton;
- A patent claim may be obvious if the claimed combination of elements was “obvious to try,” particularly when there is a design need or market pressure to solve a problem and there are a finite number of identified, predictable solutions such that a POSA would have had good reason to pursue the known options within his or her technical grasp; and
- One should be cautious of using hindsight in evaluating whether a claimed invention would have been obvious.

26. I understand that obviousness requires a POSA to have a reasonable expectation of success in combining multiple prior art references. I understand that a reasonable expectation of success refers to the likelihood of success in combining references to meet the limitations of the claimed invention. I understand that in evaluating a combination, a predictable result means not only that the prior art elements are capable of being physically combined, but also that the combination would have worked for its intended purpose.

II. DESCRIPTION OF THE RELEVANT FIELD AND THE RELEVANT TIME FRAME

27. The relevant field of the '986 patent is communications networks and methods for providing reliable, fault-tolerant network access. I have been instructed that for purposes of this IPR, I should consider the priority date of the '986 patent to be January 7, 2002.

III. BACKGROUND

A. Networking fundamentals

1. Graph theory

28. The fundamental reason for designing a network, specifically a data communications network, is to provide reliable connections between a set of devices. As part of the design of a network, a designer may choose to minimize the cost of the network considering the number of links between a set of devices, the bandwidth of the links and the financial cost of transmission over various

media. To maximize the reliability of the network, there may be a need for redundancy of communications, but this redundancy may come at a cost. In addition, it is becoming increasingly common to allow networks to support broadcasting, where information is disseminated to all of the devices connected to the network, or multicasting, where information is disseminated to a large subset of the devices connected to the network. Multicasting and broadcasting techniques are intended to simplify distribution of content across the network but may create unique problems when used with network configurations that provide redundancy.

29. An important branch of mathematics that has found its way into network design is the field known as graph theory. While graph theory can be widely applied to many diverse problems, from routing packages through a shipping network, to a salesperson finding the optimum route to visit all of their customers, to finding the best configuration for a data transmission network, the terminology used is the same – a number of nodes, sometimes called vertices of a graph, are connected via a number of links, sometimes called edges of a graph. With this general framework, it is possible to reason about the state of connectivity of a set of nodes and deal with a number of important issues. These issues include cost of connectivity, reliability of connectivity, and behavior of the system under different conditions.

30. For the absolute maximum reliability of a network, the gold standard is a fully interconnected network where every node has a direct link to every other node. Any or even most of the links in this network can fail but the nodes will continue to have a path through the network interconnecting them. Unfortunately, this fully interconnected network will be a very expensive network due to the number of links – for a network with N nodes, there on the order of N^2 links.

31. Besides the added cost of redundant links, extra links in a communications network will create the potential for loops in a network. If the network uses broadcast or multicast, these loops may cause messages to circulate through the network, creating unnecessary traffic.

32. It is a well-known theorem in graph theory that there is a structure that eliminates the potential for loops and minimizes the number of links required to connect all of the nodes. This structure is known as a tree and, like a tree in nature, has a series of limbs (links) that start at a root and provide a single path to all the branch points (nodes) and leaves (end nodes) of the tree.

33. In a natural tree, limbs may break, separating portions of the tree from the rest of the tree and disconnecting the broken part from the root and nourishment. In a similar manner, transmission links in a network using a tree structure may fail, disconnecting part of the network. To provide a reliable

network, mechanisms are needed to heal the tree, reconnecting around the failed link and reestablishing a reliable network connection to all the nodes.

34. When applying graph theory concepts to real networks, some of the terms are changed. For instance, the terms “nodes” or “vertices” defined in graph theory when applied to a network are referred to as routers, switches, and local area network (LAN) bridges. As the name implies, a local area network is a network that attaches a set of devices that are separated by a relatively short distance thus, a local area. This is in contrast to a “wide area networks” or “metropolitan area networks” that connect devices spread over a wider geographic area.

35. Unlike a tree in nature, which grows upwards from the root, in computer networking, the root is drawn at the top of the tree and branches flow downward from the root. By convention, from any intermediate node, the direction towards the root at the top of the tree is referred to as “upstream” while the direction towards the nodes at the bottom of the tree is referred to as “downstream.” The nodes at the very bottom of the tree that are not connected to any further downstream nodes are often referred to as “leaf nodes.”

36. Besides these standard networking terminology conventions, the nodes themselves are often referred to be several terms that might depend on the specifics of the network. The terms “bridge,” “switch,” “gateway,” or “router”

may be used to describe a node, but all of these terms will include devices that process messages, forwarding them to the intended destination and maintaining local information to allow efficient distribution of messages.

37. Finally, nodes have interfaces that allow them to connect to the communications links that actually convey data. These interfaces are referred to as “ports.” Other than a leaf node that only connects to one other node in the network, each node will have at least two and generally more than two ports to allow interconnection to a set of other nodes in the network to create the tree.

2. Spanning tree

38. Just as a forest may have a number of separate trees, a network may have a series of disconnected segments. To enable connectivity between all the nodes of the network, a single tree that “spans” all the nodes is desirable.

“Spanning” a set of nodes in networking parlance means to connect that set of nodes. In network terminology, a set of connections that spans a set of nodes is called a spanning tree.

39. For most networks, there may be several choices of how the nodes may be interconnected, allowing alternate arrangements of links to provide the desired connectivity. This is where the cost of operating the network enters. Some links may have a lower financial cost or may provide faster connections. If an effective cost to the end user is attributed to a lower speed connection, it may be

desirable to minimize the overall cost of using the network. A spanning tree that connects all the nodes while minimizing cost is called a minimum cost spanning tree or a minimum spanning tree. As discussed later, IEEE standardized a protocol to find and maintain the minimum spanning tree for a given network in the presence of failures within the network. This IEEE standardized protocol is called the Spanning Tree Protocol (STP).

40. The spanning tree, by its nature, connects all the nodes in the network and the STP optimizes the cost of connectivity across the entire spanning tree. This might come at an increased cost of operating the network when failures occur and the entire network must be reconfigured, but approaches have been designed to minimize the amount of adjustment needed by confining reconfiguration alerts to nearby nodes in the network. This is referred to as “localizing” alerts or minimizing the “radius” of alerting and will be described below.

3. Network protocols

41. Data networking protocols underly all the data communications we depend on. In order to provide structure to understand the technology, the International Organization for Standardization (ISO) published a seven-layer protocol reference model “stack” to divide a complex problem into smaller, more

manageable pieces.¹ This is sometimes referred to the Open System Interconnection (OSI) basic reference model. The OSI reference model is generally represented as a diagram as shown in Figure 1.

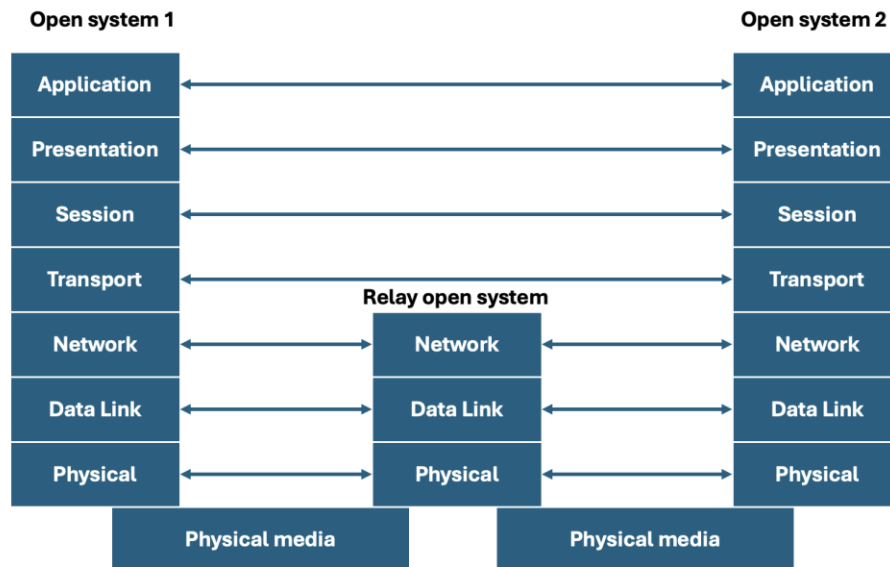


Figure 1 - OSI reference model

42. While the OSI reference model was first used to describe networks in international standards, such as X.25, its concepts have been applied to other networking standards that do not necessarily follow the same strict separation of functions. For instance, the Internet is built on Transmission Control Protocol/Internet Protocol (TCP/IP) which mixes some of the functionality of

¹ ISO/IEC 7498-1:1:1994 Information technology - Open System Interconnection - Basic Reference Model: The Basic Model.

layers as shown in Figure 2. Nevertheless, it is instructive to examine the different functions of the various protocol layers to understand the operation of typical networking equipment. In the OSI and the TCP/IP model, the physical layer, sometimes referred to as the PHY level, addresses the physically tangible elements of communications, that is the connectors, copper wires and electrical voltages used for communications. The next higher level of both protocol models is sometimes referred to as the media access control (MAC) layer, where signaling access to the physical media is controlled. Because of the overlap of functionality between the OSI and TCP/IP models at the bottom of the TCP/IP Network Access layer, some of this functionality would be considered part of the OSI Physical layer and some part of the OSI Data Link layer.

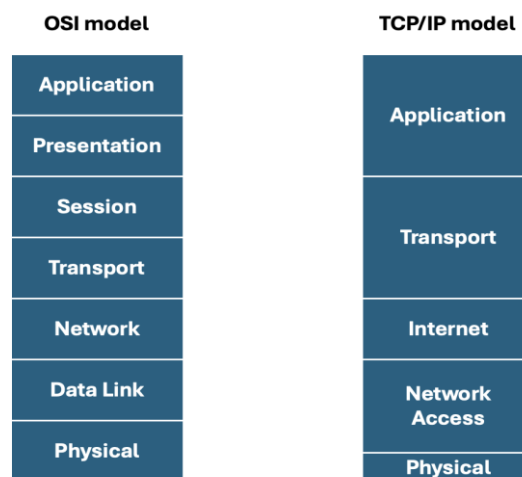


Figure 2 - OSI versus TCP/IP correspondence

43. Specifically, when interconnecting routers, switches, and LAN bridges, there are various functions which are performed at different levels of the

protocol stack. Some layers must account for information that is widely shared across large expanses of the network. For instance, the IEEE 802.1D Spanning Tree Protocol (STP), as well as its variants, operates at higher levels of the protocol stack, coordinating interconnection between multiple network elements.

44. The STP continually sends messages through the spanning tree to continually verify operation of the network. In many systems, these messages are known as “heartbeat” messages that can be used to see that the system is still “alive.” STP uses a Hello BPDU message, defined below, that is sent periodically, every few seconds by default, although this period can be modified by the network operator. A bridge failure detected by the STP, that is, a period of missing Hello BPDU messages, needs to cause alerts to all the devices on the network that are depending on the spanning tree the STP controls. In contrast, since a network switch acts as a relay open system in the OSI model, it may detect a link failure at the Physical layer of the protocol stack. In this manner, IEEE 802.3 defines a set of procedures to detect and respond to link failures before the upper layers of the protocol stack will be aware of disruption of the network.

45. Device and link failures are generally detectable at various levels of the protocol stack as the required interactions are interrupted, either loss of communications over a physical link or failure to receive STP messages. Since all bridges, switches, and routers are connected using IEEE 802.3, all devices are

capable of detecting link failures when the connection stops responding. In most network designs, these lower-level functions operate more frequently, allowing earlier failure detection than at higher levels. For instance, the IEEE 802.3 physical level link integrity checks typically are made every 10 milliseconds while the higher level IEEE 802.1D Hello BPDUs status checks are typically performed every two seconds. More frequent status checks allow faster failure detection and faster recovery from any faults in the network. On the other hand, link failure detection only alerts the nodes connected to the failed link, requiring the nodes to alert others to the failure. Higher level failure detection, that is at the MAC, Internet, or Transport level, will often inform a broader set of nodes of a failure, albeit more slowly. Network devices generally use multiple simultaneous failure detection techniques to ensure the highest probability of failure detection. Besides the speed of response of some techniques, detection of a failure of a physical communications link allows bidirectional failure detection, providing information to devices upstream and downstream of the failure as described later.

B. Setting up a Spanning Tree Network

46. The IEEE 802.1D-1998 Spanning Tree Protocol (STP) standard, published in 1998, defines the procedures and signaling messages required to initially establish a minimum spanning tree across a network of communications

devices. Networks that operate in accordance with the IEEE 802.1D-1998 standard are sometimes referred to as “IEEE 802 networks”.

47. The bridges and switches used in data networks, such as IEEE 802 networks, are interconnected to allow information to be distributed reliably. To do this, all the devices need to have a consistent view of the network interconnectivity, even as the interconnectivity may change due to newly added equipment or node or link failures. The IEEE 802.1D-1998² standard defines a self-healing protocol, known as STP, to create and communicate and maintain information about a “spanning tree,” that is, a series of interconnected nodes that first, span all the nodes and second, does so creating a tree structure.

² Throughout this discussion, where a specific standard is identified, the date of the standard will be included, e.g., 1998 for IEEE 802.1D. This is because standards evolve over time. Later in this discussion, improvements to IEEE 802.1D-1998 (Spanning Tree Protocol) are identified such as IEEE 802.1W-2001 (Rapid Reconfiguration). IEEE 802.1W-2001 was superseded by IEEE 802.1D-2004 when the former standard’s features were incorporated into the latter standard, thus distinguishing the 2004 version of 802.1D from the 1998 version. If no date is associated with the standard, the features common to all versions of the standard are implied.

48. It is necessary to “span” all the nodes, i.e., routers and switches, to ensure that all nodes are interconnected, that is, none are “left out” of the assemblage lest they be unable to communicate. A tree structure is needed to ensure that there are no loops in the communications network. Loops are to be avoided particularly when broadcast or multicast traffic will be sent over the network. If any loops exist, there is the possibility that, for instance, broadcast messages might circulate around a loop, creating unnecessary network congestion. A well-known so-called “broadcast storm” occurs if a broadcast message is continually rebroadcast around a loop in a communication network, unnecessarily bogging down communications. Network designers have long known this is easily eliminated by using tree structures to avoid loops. A fully connected tree structure, that is, one that spans all the nodes, will ensure all nodes are linked with no loops in the network. A tree also allows complete interconnection of the nodes with the minimal number of links. It is often the case that there will be alternate choices that can be made about which specific links in the network are to be included in the tree. This selection is generally made based on cost. While cost could have a monetary component, in most networks, even if there is no direct monetary cost associated with the choice of one link over another, there will be an operational cost difference. For instance, a lower speed link will take longer to transmit the same message than a higher speed link would, so a higher cost can be assigned to

the lower speed link in assessing the overall cost of one tree structure versus another.

49. While the connectivity of a tree structure is needed in the network to ensure the absence of loops, a tree is an inherently unreliable structure. A failure of any single link in the network will sever portions of the network that are connected through this link into two distinct, separate networks with no ability to communicate. For this reason, IEEE 802.1D-1998 defines recovery mechanisms to reconnect the portions of the network that would be separated by a link failure.

50. Figure 3 illustrates a network with a series of interconnected nodes having a variety of interconnection links without regard to a spanning tree. There are a number of potential routes from any node to any other node and many possible loops within the network. Although it is not apparent in this figure, one of the nodes could be considered a “root” node from which all connectivity in the network emanates. The root node may be manually assigned where there is some reason to do so or the network can self-identify the root node. For this example, Node A will be considered to be the root node of the network.

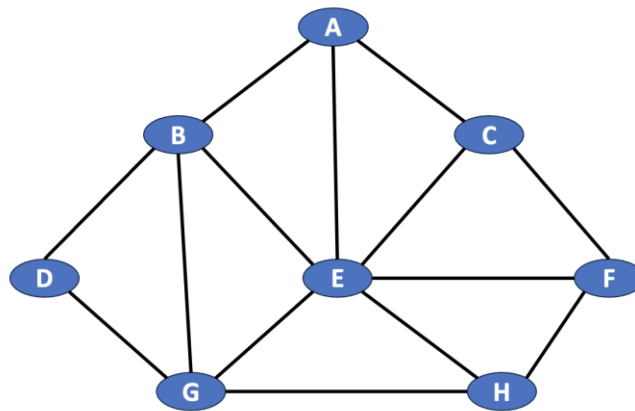


Figure 3 - Original network

51. Figure 4, in contrast, shows the same set of nodes with a subset of the links active (the links shown with thick black lines) while many of the links have been deactivated (the links shown with thin grey lines). This is the same network as shown above but only the active links are used for communications. As indicated earlier, the topmost node in the network, Node A, has been selected as the “root node” of the network and all other nodes have been connected in a tree originating from the root node.

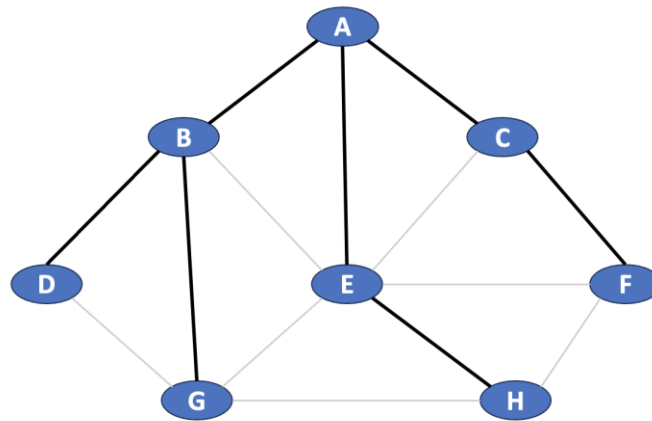


Figure 4 - Network tree

52. Figure 5 shows one possible failure condition where the network of Figure 4 has suffered a link failure (indicated by the red X) on the link connecting Node A to Node E³. Link AE has been marked as a thick dashed line indicating that it was an active link but is now out of service. After this failure, there are now two separate trees, one consisting only of Nodes E and H and a disjoint tree with the rest of the nodes in the original network. Without reconfiguration, Nodes E and H can communicate with each other, but cannot communicate with the rest of the nodes in the network.

³ Using the standard convention in graph theory, a link is identified by the nodes it connects, so the link between Node A and Node E will be commonly referred to as Link AE.

54. While the term “nodes” is commonly used in graph theory, for the rest of this discussion, the term “bridge” will generally be used to refer to the nodes, as “bridge” is a generic term for network components such as switches. A switch is a type of bridge.

55. Through a series of message exchanges described below, the IEEE 802.1D-1998 Spanning Tree Protocol automatically discovers the failed link and enables a replacement link to become active, ensuring automatic restoration of the network connectivity while maintaining the loop-free tree configuration.

1. STP Bridge Roles

56. As mentioned above, a network tree grows downwards from a root node. In the IEEE 802.1D-1998 standard, the node at the root of the tree is called the Root Bridge and has the special function of distributing control signals through the network. For instance, if a node interior to the network detects a failure, for instance because the periodic Hello BPDU message has not been detected for a specified length of time according to the STP standard or if it detects a failure of a link integrity check via the IEEE 802.3 standard, the node will transmit a message upstream to the Root Bridge alerting it of this failure. The Root Bridge, in turn, rebroadcasts a message to all the downstream nodes in the network, alerting them to the failure.

57. The first step of the IEEE 802.1D-1998 Spanning Tree Protocol (STP) is to determine the root bridge, the “master node” in the network that all other bridges need to be connected through. In some instances, this can be performed automatically by the network using the STP or if appropriate, the root bridge can be set manually by the network administrator. The root bridge is determined using the Bridge ID value (BID). The bridge having the lowest numerical value is the one that will be selected as the root bridge. Of course, the network administrator can predetermine which bridge will have the lowest BID by setting the value or allow the network to use the STP to discover this on its own. It is possible that two bridges may have the same BID, in which case the Media Access Control (MAC) address of the bridges’ ports will be used and the bridge with the lowest port MAC address will be used as the root bridge. Since MAC addresses are unique to bridges and their ports, it is guaranteed that there will be no duplicate MAC addresses and one will be lowest valued.⁴ Figure 7 illustrates a simple network

⁴ MAC addresses are 48-bit quantities used to uniquely identify network elements in IEEE 802 networks. While they are generally represented as 6 groups of two hexadecimal digits, for simplicity in the following illustrations two-digit numbers will be used. Each port on a bridge has a unique MAC address and the bridge itself has a unique MAC address. Similarly, while BID values are 64-bit quantities

with bridges labeled with their BID and MAC addresses as well as port MAC addresses and the costs associated with links. Note that the label “M” is used in this and the subsequent figures to indicate a MAC address. It should be appreciated that the MAC address of a switch is marked inside the node, e.g., “M=90” while the port MAC address is marked outside the node, e.g., “M=57.” In addition, the abbreviated MAC addresses in the range less than 90 are used to label port MAC addresses while abbreviated MAC addresses equal to or greater than 90 are used to label switch MAC addresses.

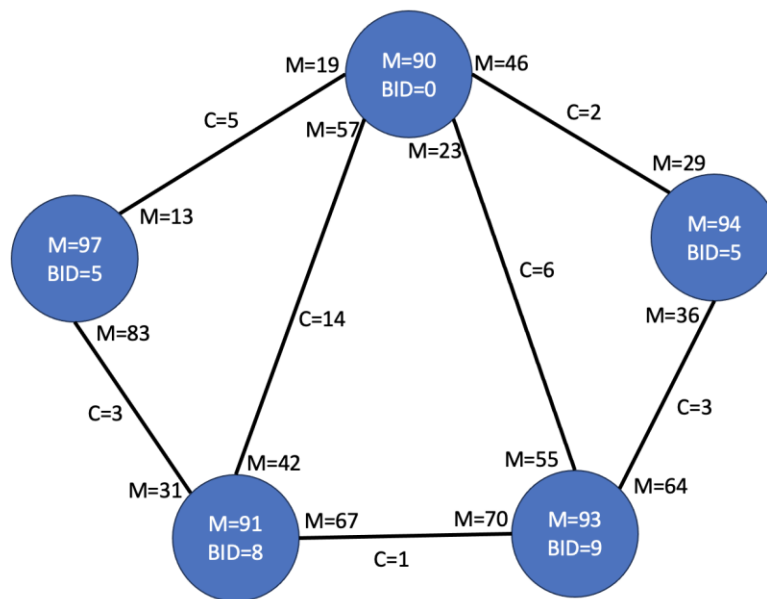


Figure 7 - Simple network to illustrate STP

representing a 16-bit priority plus the MAC address, a one-digit number will be used for the priority combined with a two-digit bridge MAC address in the range of 90-99 will be used for illustration.

58. Whether a root bridge BID is set manually to ensure the desired root bridge will be found by the network or the network is allowed to find its own root bridge, the root bridge selection begins with all bridges multicasting a “Hello BPDUs”⁵ message, essentially having each node nominate itself as the root bridge. The message contents indicate that each sending node is calling itself the root bridge and it has a zero-cost route to the root node. The format of the BPDUs message is shown in Figure 8.

Prot ID	Prot ver	BPDUs type	Flags	Root ID	Root path cost	Bridge ID	Port ID	Msg age	Max age	Hello time	Fwd delay
16 bits	8 bits	8 bits	8 bits	64 bits	32 bits	64 bits	16 bits	16 bits	16 bits	16 bits	16 bits

Figure 8 - BPDUs format

59. Each bridge compares its own Hello BPDUs message to the messages it has received. If a bridge sees that the received Hello BPDUs messages have a higher bridge priority value than their own, they discard it, knowing that the other bridges can’t be the root bridge. Each bridge only retains the Hello BPDUs messages that have a lower value for bridge priority than theirs and sets the root bridge ID to the value of the winning bidder. Each bridge also calculates the path

⁵ BPDUs is a Bridge Protocol Data Unit, the message used to detect loops in the network and convey other information about the network.

cost, that its cost of networking from the bridge to the root bridge by adding the path cost sent by the root bridge to the path cost associated with the link it has connected to. This modified Hello BPDUs message is then forwarded from each bridge to other bridges in the network. Each bridge compares all the Hello BPDUs messages it receives and stores the one with the lowest path cost. Finally, using the path cost comparisons, the lowest cost paths are kept and their ports marked as designated ports while the higher cost paths are blocked by marking the ports connected to those paths as blocked ports.

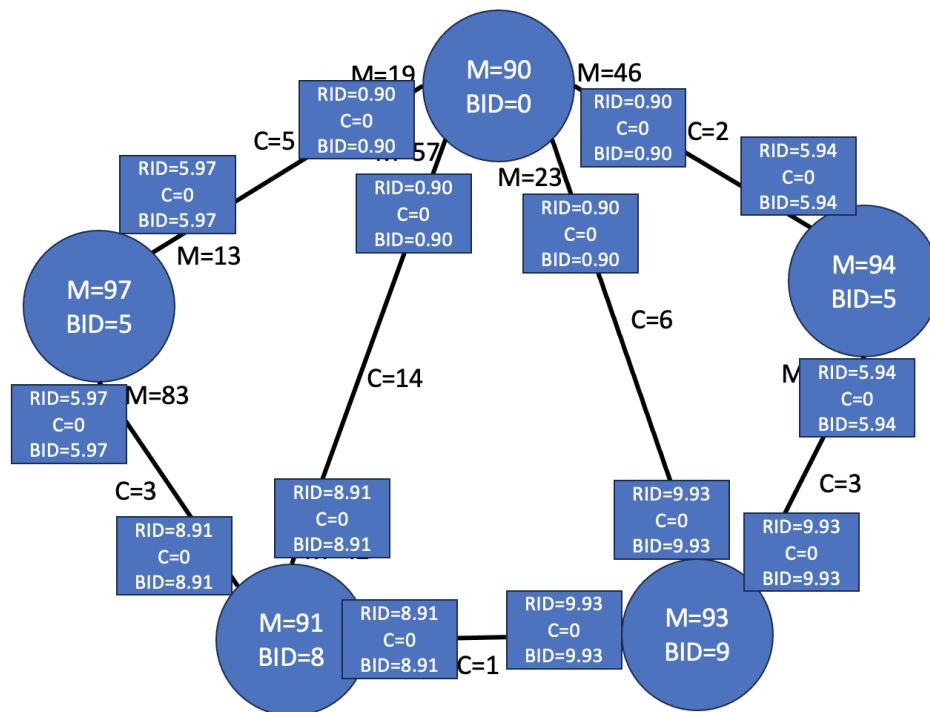


Figure 9 - Hello BPDUs broadcast for initialization

60. Figure 9 shows a series of Hello BPDUs messages (represented by the rectangles next to the bridge sending the message) sent from each bridge in the

network indicating that they are the root bridge, showing a cost of zero for their connection and providing their bridge ID. As described previously, there are two parts to the bridge ID: (1) a 16-bit bridge priority, which can be set by the network administrator, allowing them to influence which bridge will become the root bridge and (2) the 48-bit bridge MAC address, which is unique to each device. Thus, the device at the top of Figure 9 transmits a value BID=0.90, concatenating the bridge priority 0 with the bridge MAC address 90. Similarly, since each bridge initially claims to be the best root bridge, it transmits its own bridge ID as the root bridge ID (RID) so, for instance, the device at the bottom right of Figure 9 transmits BID=9.93 and RID=9.93. In this figure, the box next to the node is the BPDU message sent from that node to the adjacent node. In this example, since each bridge is connected to the bridge with MAC address 90 (M=90), each bridge can verify it has a higher priority value than that bridge and therefore relinquishes the root bridge bid to the bridge with MAC address 90. Further, each bridge can determine its route cost to the root bridge from the link cost. The resulting Hello BPDU messages with the elected root bridge are shown in Figure 10.

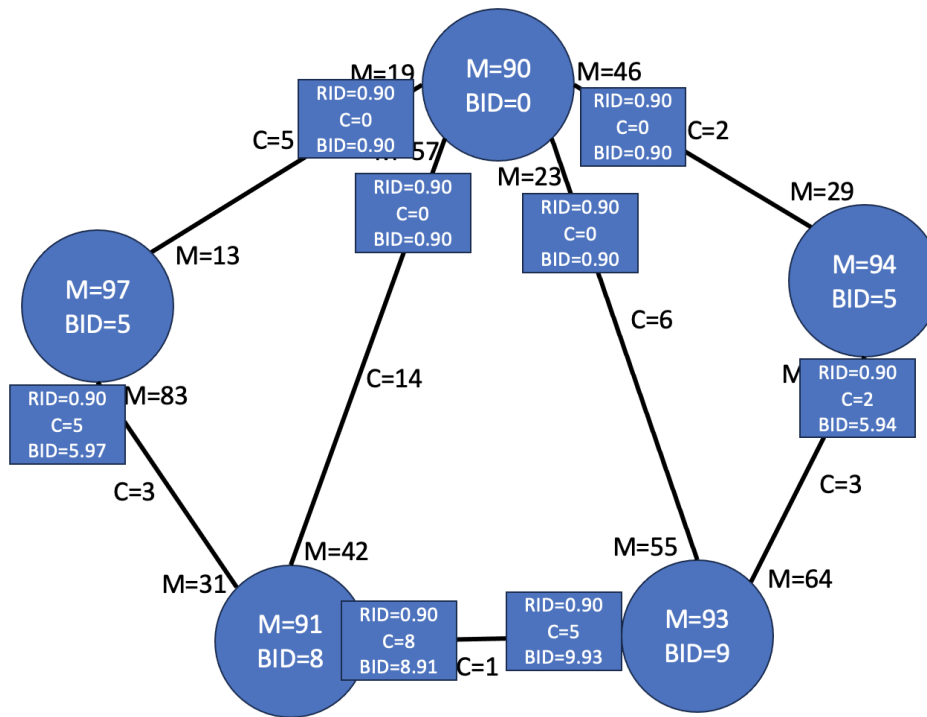


Figure 10 - BPDUs messages after root bridge election

61. With the root bridge identified, the least cost route from each node is found and the spanning tree can be identified. This is shown in Figure 11. The links with solid lines are the links chosen to be included in the spanning tree while the links with dashed lines are not part of the spanning tree. As an example, consider two of the links from the root bridge, M=90: first, the link from Bridge 90 to Bridge 97, having a port MAC address at Bridge 97 with a value M=19 is connected to the port at bridge 97 having a port MAC address M=13 is an active link; second, the link from Bridge 90 to Bridge 91, having a port MAC address at Bridge 97 with a value M=57 is connected to the port at Bridge 91 having a port MAC address M=42 is an inactive link. The roles of these four ports in the STP

will be described in the section below, reflecting the status of the active and inactive links.

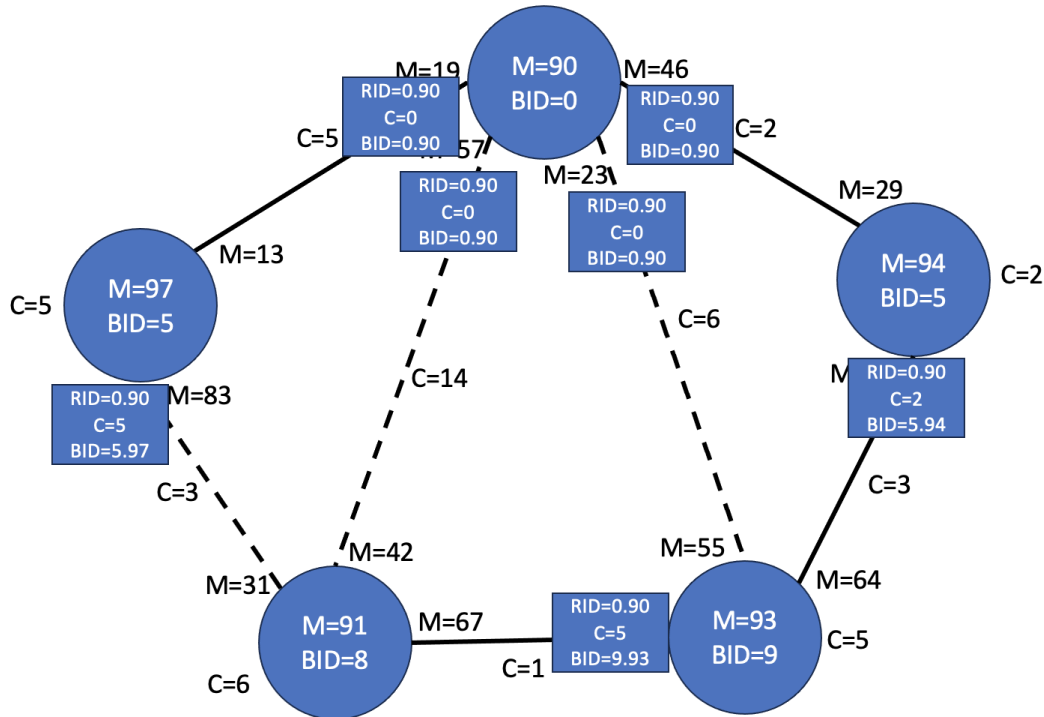


Figure 11 - Network with least cost spanning tree identified

2. STP Bridge Port Roles

62. Since each node has multiple ports to connect to different links, the role of the various ports must be noted. Any port on a bridge that provides the least cost path to the root bridge is called the “root port.” Each bridge may have only one root port. At the other end of a link where it connects to a port on the next bridge, that next bridge’s port is called the “designated port.” This process is repeated for each node. A node that is not a root node can have multiple

designated ports. When complete, all the ports connected to the unused links in the network are marked as either “blocked ports” or as designated ports depending on the implementation. The STP and RSTP standards identify the two above-noted port roles (root and designated) that are central to the operation of the spanning tree: The RSTP standard also identifies two additional important port roles: alternate ports, providing an alternate upstream root port if the original root port fails and backup ports, which can be used as downstream replacements for failed downstream designated ports. While the standards address necessary conditions to provide a spanning tree and avoid network loops, like most standards, they do not prescribe all the operating conditions for the network. Specifically, ports that are not actively sending and receiving traffic as part of the spanning tree may have options governing their operation. For example: (1) all downstream ports may operate as designated ports, sending traffic that may pass through the bridge or (2) only those ports that are part of the spanning tree may operate as designated ports, connected to the root port on the next bridge and actively sending traffic through the spanning tree. The network equipment designer and network administrator may have reasons to consider each of these options. In the second case, the networking load on the bridge is minimized since it does not need to copy data streams to ports that will not have reason to send the information – no one is listening to the link the port is connected to lest loops be created in the network.

On the other hand, in the first case, sending the traffic from the ports that are not actively in use as part of the spanning tree could make network restoration faster since the downstream bridge need only switch its inactive blocked port into an active state to join the spanning tree when needed. This is, in essence, the function of alternate ports in the RSTP standard, but is a design choice for any network.

63. Figure 12 shows the example network with the role of each port marked on each bridge as R (root port role), D (designated port role), or B (blocked port role).

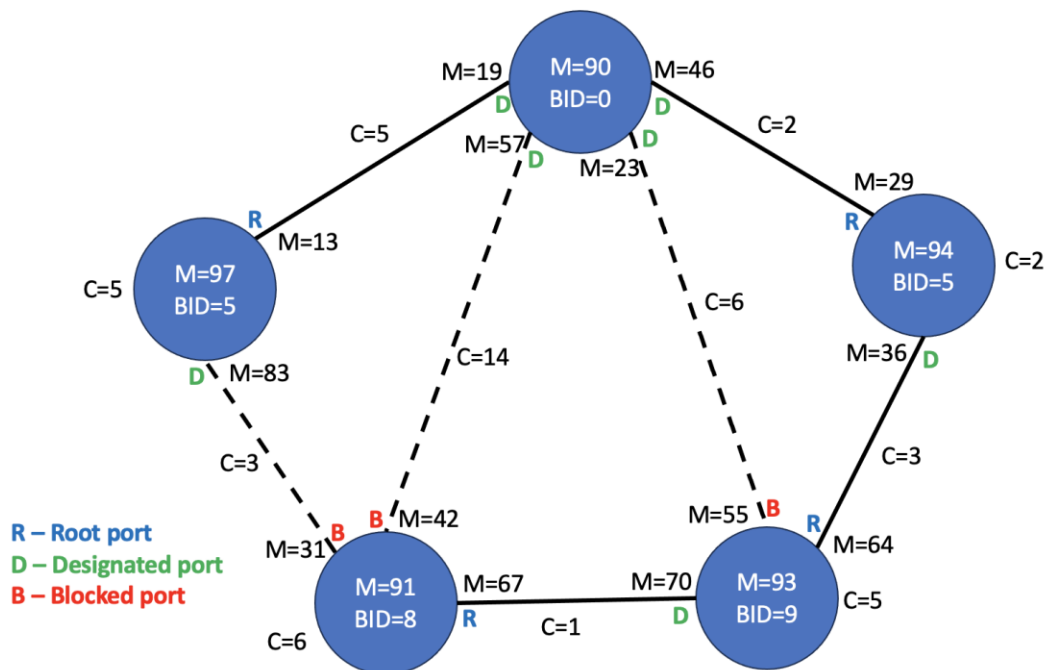


Figure 12 - Network with spanning tree and port status marked

64. Ports that could create potential loops in the network if they were active are placed into a blocked status so that they ignore any messages received

but remain prepared to take over if a new network element changes the configuration of the network or a link failure occurs.

65. Once the loop-free spanning tree has been determined and the network designating the status of all the ports, the root bridge broadcasts Hello BPDU messages over the network every two seconds to ensure the network has remained connected and no new nodes are present.

66. Another illustrative figure, which I understand is being used in the Petition in this matter, follows below. Labeled Figure 13 in this Declaration, it shows a computer network prior to executing the STP algorithm. The network is formed by connecting bridge A to bridges B and C via their respective ports. Bridge A is connected to bridges B and C via redundant links, which creates a loop. The loop's existence is not in accordance with STP, which removes such loops from a network.

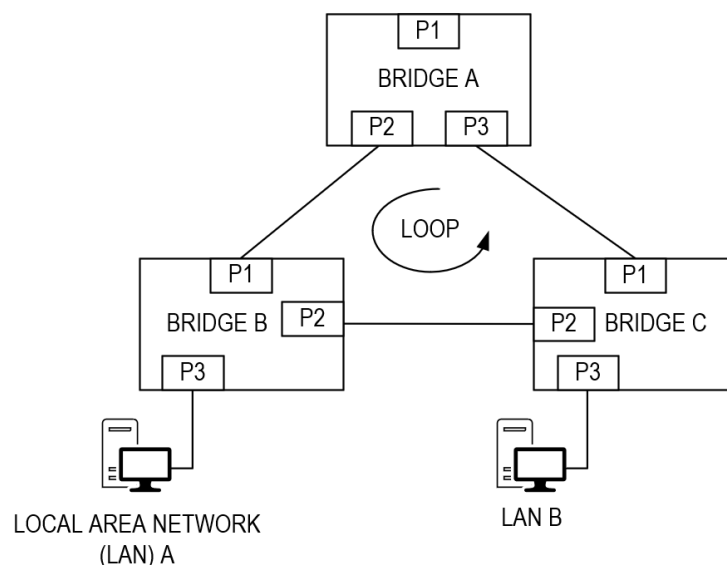


Figure 13 – computer network with loop; not in accordance with STP

67. Another illustrative figure, which I understand is being used in the Petition, follows below. Labeled Figure 14 in this Declaration, it shows the result of applying STP to Figure 13 above. The STP algorithm forms the tree structure shown in Figure 14 below by first “spanning” the nodes to ensure they are all interconnected. For example, each node (i.e., bridge or LAN) is connected to at least one other node in the network. Second, the STP algorithm creates the tree structure by identifying a root bridge at the “top” of the tree and “branches” and “leaves” downstream of the root bridge. In Figure 14 below, the STP algorithm identifies designated bridge A and designated bridge B as the “branches” of the tree and local area network (LAN) A and LAN B as the “leaves” of the tree. A bridge (e.g., root bridge or designated bridge) is a switch that connects two LANs. For example, designated bridge A is a bridge because it connects LAN A and LAN B (via designated bridge B).

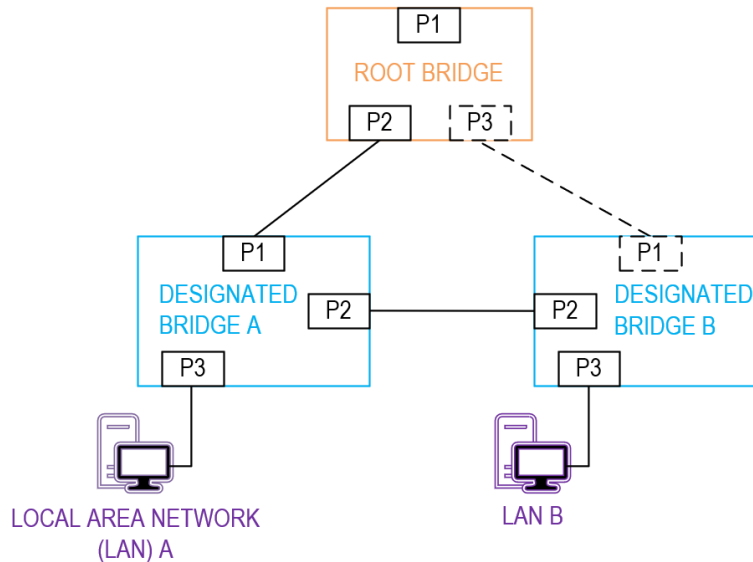


Figure 14 – computer network in accordance with STP

68. Once the tree structure is created, the STP algorithm manipulates the states of nodes' ports to eliminate loops. As represented by the dashed line between the root bridge and designated bridge B in Figure 14 above, the STP algorithm blocks port P3 of the root bridge and port P1 of designated bridge B to eliminate the connection between them, thus eliminating the loop that previously existed in the network (shown in Figure 13 above). However, other implementations also comply with the IEEE 802.1D standard, such as making port P3 of the root bridge a designated port in the forwarding state. If port P3 of the root bridge remains in the forwarding state, the link shown with a dotted line remains inactive because port P1 of designated bridge B is blocked. Thus, the Hello BDPUs that are sent along that link from port P3 of the root bridge to port P1 of designated bridge B are discarded (dropped). While the network can

be configured to keep downstream ports such as port P3 of the root bridge in a forwarding state when a link to such port is inactive, POSAs had reason to block such downstream ports, namely to save the resources involved in preparing and sending messages that will be ignored when received on a blocked port at the other end of an inactive link. Thus, both approaches are (and were, before the '986 patent) possible and obvious: a downstream port can be left in a forwarding state when the corresponding port on a switch at the downstream end of an inactive link is blocked, which would offer a small simplification by avoiding the need for changing the state of such downstream port; or, a downstream port can be changed to a blocking state when the corresponding port at the downstream end of an inactive link is blocked, which offers extra efficiency in avoiding the need to prepare and send messages (such as regular Hello BDPU messages) that will be ignored when received on a blocked port of a downstream switch.

69. The STP algorithm operates by assigning bridges a role: root or designated. The STP algorithm identifies one and only one bridge in the network as the root bridge and identifies the remaining bridge(s) as designated bridge(s). A root bridge is a reference point for all other bridges in a network having a spanning tree topology. A designated bridge routes its traffic upstream toward the root bridge. A LAN connects to a designated bridge. In Figure 14 above, the STP algorithm identifies bridge A of Figure 13 above as the root bridge and bridges B

and C of Figure 13 above as designated bridges A and B, respectively. LANs A and B are connected to designated bridges A and B, respectively.

70. The result of assigning roles to the ports of the bridges in Figure 14 above is shown in Figure 15 below. A bridge's root port is used to send data messages to the root bridge along the lowest cost path, i.e., the path with the fewest number of hops or stops in a network from a designated bridge to the root bridge.

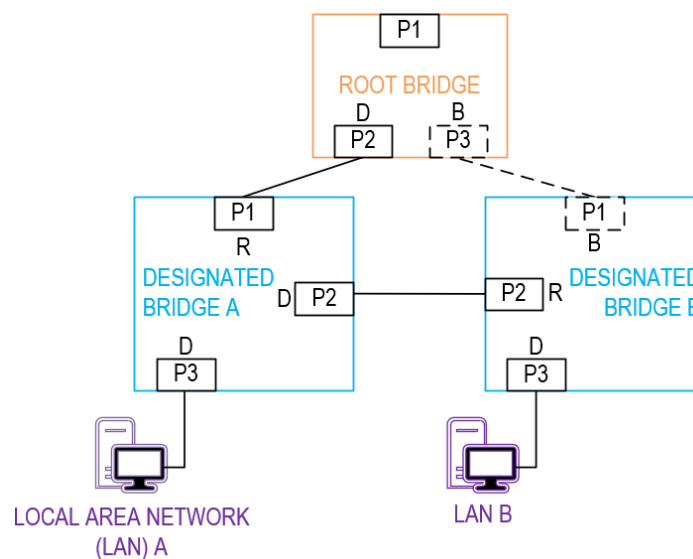


Figure 15: Roles of bridges and their ports in STP

71. Ports P1 of designated bridge A and P2 of designated bridge B of Figure 3 above are root ports (“R”). Ports P2 of the root bridge, P2 and P3 of designated bridge A, and P3 of designated bridge B are designated ports (“D”). A designated port can connect to a LAN or to another bridge's root port. Ports P3 of the root bridge and P1 of designated bridge B are alternate or backup ports, called “blocked ports” (“B”). A backup port receives but does not forward data

messages, thus blocking data messages from being forwarded therethrough. However, it would still comply with the IEEE 801.2D standard to make port P3 of the root bridge a designated port, which continues to forward messages. As I discussed above, there is no practical difference in terms of network communication between leaving port P3 of the root bridge in a forwarding state versus moving it to a blocked state, since any messages it sends over an inactive link to blocked port P1 on designated bridge B will be discarded in any event. For the reasons I have already explained, it would have been obvious to place port P3 of the root bridge in a blocked state to avoid unnecessary effort and burden involved in preparing and sending messages that will be discarded upon arrival at a blocked port P1 of designated bridge B in any event.

3. STP Bridge Port States

72. In addition to the three roles that a port may have, that is root port, designated port or blocked port, the IEEE 802.1D-1998 standard also defines various states that each port of a bridge may be in during the operation of a network. The possible roles of a port are summarized in Table 1 below. Table 2 summarizes the possible states of a port.

Port Role	Function
Root Port	The bridge port connected to a link with the least cost path to the Root Bridge
Designated Port	The bridge port connected to the link that is connected to a Root Port on the next bridge

Blocked Port	A port that is blocked from forwarding traffic, preventing network loops
--------------	--

Table 1 – STP Port Roles

Port State	Function
Forwarding	The port will process BPDU messages, receiving and forwarding frames
Learning (transitory)	The port learns about BPDUs
Listening (transitory)	The port listens to BPDUs
Blocking	The port receives BPDUs but drops (ignores) them
Disabled	The port is not involved in the STP

Table 2 – STP Port states

73. In Figure 15 above, the root ports and the designated ports are in the forwarding state.

74. As Table 2 indicates, the possible states of a port are Forwarding, Learning, Listening, Blocking, or Disabled. Other than the Disabled state, which means that a port does not participate in the STP, ports will normally progress through the other four states as network configuration, failures, and reconfigurations occur. Ports that are in the Forwarding state process BPDU messages, receiving and forwarding them. Ports that are in the Blocking state may receive BPDUs but ignore or drop them. The Learning and Listening states are transitory states used during recovery from a failure where a port acquires information about the connectivity of the network to allow restoration of

connectivity after a link failure or other change in network connectivity. If a port in the Forwarding state stops receiving Hello BPDU for a specified time (20 seconds according to the STP standard), it transitions into the Listening state, looking for a connection through one of the unused links. The port remains in the Listening state for a specified time (15 seconds, by the standard default) and transitions to the Learning state, where it learns the MAC addresses of connected devices. After a specified time (another 15 seconds by default), the port transitions to the Forwarding state and the connection to the Root Bridge is restored. This process is illustrated in Figure 14 below.

75. In Figure 15 above, ports P3 of the root bridge and P1 of designated bridge B are in the blocking state (though, as discussed above, that need not necessarily be the case for port P3 of the root bridge). Ports in the blocking state are designated as back-up ports. Upon detection of a failure at a port in the forwarding state, one of the back-up ports can be transitioned from the blocking state to the forwarding state.

4. Maintaining STP Using Bridge Databases

76. In order to control the operation of the network and the included bridges, STP uses a number of databases in each bridge. For instance, the IEEE 802.1D-1998 standard specifies the Filtering Database in Sections 7 and 14 of the standard. As an illustration, Figure 6 below shows how the filtering database

instructs the bridge to forward data frames, based on the state of the port and the addressing of the data. Similarly, Figure 137 below illustrates how, during the Learning state of a port, data is monitored to learn about network connectivity.

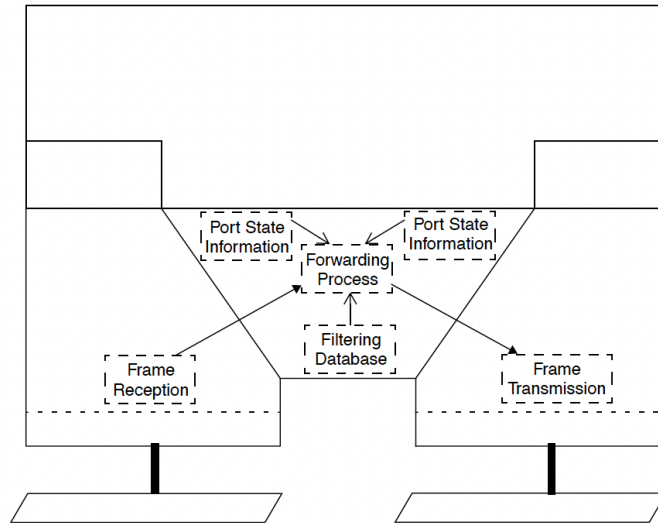


Figure 16 - IEEE 802.1D-1998 Standard "Figure 7-4 Relaying MAC frames"

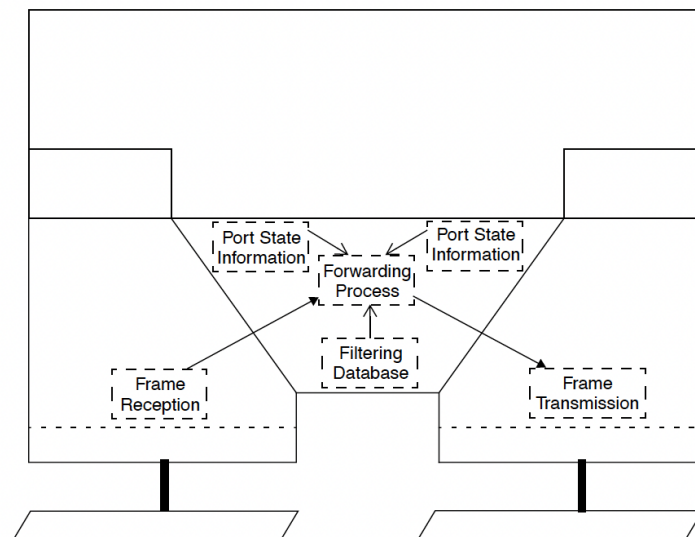


**Figure 137 - IEEE 802.1D-1998 Standard
"Figure 7-5 Observation of network traffic"**

77. The IEEE 802.1D-1998 standard describes how the database learns from signaling messages and uses this information to control the operation of the network, ensuring orderly processing of data in the face of failure and reconfiguration.

78. As shown in **Figure 6**, when a frame arrives at one port of a bridge, for instance the left-most port, it is received by the Frame Reception process. The frame is sent to the Forwarding Process and based on the Port State Information, will be processed if appropriate. The Filtering Database is used to decide if the bridge has information to relay the frame to the next bridge in the network, based on the destination MAC address of the frame. While only one other port (the right-most port of the figure) is shown, generally a bridge may have multiple ports available to forward the frame and the filtering database will indicate which port the frame should be forwarded to. Since the arriving frames were routed to a bridge based on the filtering database in another bridge, only frames that can be routed to their destination should arrive at the input port. There are two types of exceptions: broadcast or multicast frames are, by their nature, to be distributed everywhere in the network so filtering is not needed; and frames that have been received in the presence of a network failure. Where a network failure has occurred, a new path through the network needs to be learned and the STP

recovery method will be used, where the bridge listens to traffic and learns of new connectivity.

79. When a bridge receives a data message, it looks up the destination address of the data message in its filtering database. If a bridge receives a data message having a destination MAC address that is already present in its filtering database, the bridge forwards the data message using the bridge's port identified in the filtering database that corresponds to the destination MAC address. If a bridge receives a data message having a destination MAC address that has not been previously recorded in the filtering database, the bridge stores this new destination MAC address in its filtering database and, in combination with knowledge about which port this message was received, creates a new record in the filtering database to include a new mapping between the received port and the source MAC address. In this manner, the bridge can use the new mapping when it receives future data messages from the destination directed back to the source of the data message. Additionally, the bridge forwards the data message to the next bridge in the spanning tree, which causes that receiving bridge to similarly update its respective filtering database in the same manner as the transmitting bridge.

C. STP Self-Healing Process

80. Link failure can isolate portions of the network. To mitigate the effects of a link failure, the IEEE 802.1D standard defines STP to carry out self-

healing recovery mechanisms to reconnect portions of the network that are disconnected by a link failure. In general, an STP bridge recovers from a link failure detected at its root port by selecting a different port, such as a blocked port, to be a candidate new root port. STP causes the previously-blocked port to transition through states (i.e., from the blocking state to the forwarding state) and, if successful, selects the previously-blocked port to be the new active root port. Network operation resumes by the bridge receiving and forwarding data messages through the new root port.

81. The root bridge continuously sends out Hello BPDU messages so that downstream bridges can monitor the health of the network. The rate at which Hello BPDU messages are sent is adjustable but the default value per the IEEE 802.1D-1998 standard is every 2 seconds. Root ports on each bridge receive these messages and forward them to all of the bridge's Designated ports where they are resent. Blocked ports also receive Hello BPDU messages over the link from the previous bridge's Designated port, but they drop (ignore) them.

82. The port roles, described above, are Blocked port, Root port, and Designated port. The port states, also described above, are Forwarding (associated with a Root port or Designated port), Blocked (associated with a Blocked port), Listening (a transitory state), and Learning (another transitory state). There is also a Disabled state for ports not involved in the STP.

83. If a Designated port does not hear Hello BPDU message for the Max Age timeout value, a settable parameter with a default value of 20 seconds, it transitions into Listening state. It then listens for the Forward Delay time out value, another settable parameter with a default value 15 seconds, then transitioning into the Learning state, again for the for Forward Delay timeout value. The port then becomes a Root port. Figure 148 illustrates the state transition diagram that has just been described. The overall delay from the last received successfully received Hello BPDU message until the reconfiguration is a total of 50 seconds (20+15+15). In larger networks, this can be a significant delay with the large number of bridges and links with the possibility of any link failure increasing due to the sheer number of links.

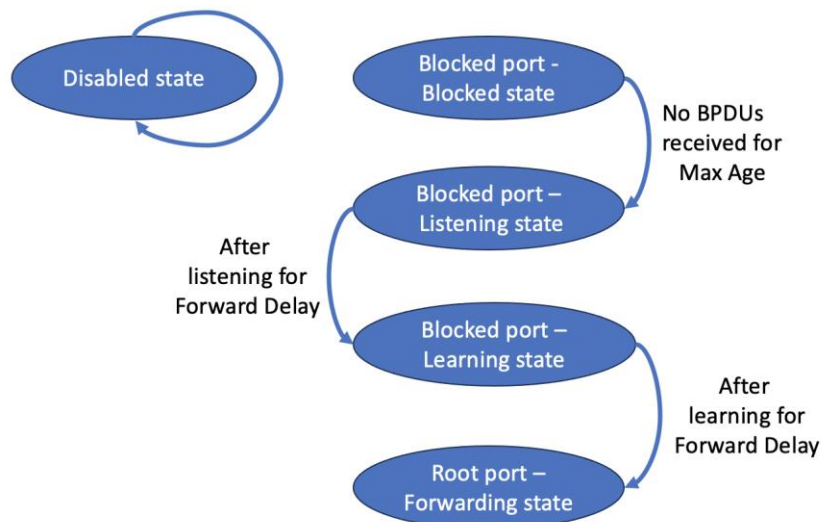


Figure 148 – IEEE 802.1D-1998 state transition diagram

84. The description above indicates how a network uses the STP to deal with network failures. In addition, the STP can detect changes to the configuration of the network. In addition to using the Hello BPDU messages, the STP also defines TCN BPDU and TC BPDU messages. A bridge detecting a network topology change sends a topology change notification (TCN BPDU) message upstream to the root bridge which then broadcasts a topology change (TC BPDU) message downstream to the entire network, alerting all the bridges of the network topology change.

1. Link Failure Detection

85. STP directly incorporates IEEE 802.3 Part 3-1998⁶ since it relies on the physical level links that interconnect bridges. Every 10 milliseconds, a bridge sends a physical level message over the communications link to ensure the link is operating properly in both directions. In addition, the connection is continuously monitored to detect if the proper voltages are present to ensure a physical electrical connection has not been lost. U.S. Patent No. 6,032,194 (“Gai”) illustrates one way how link failure detection can be used: “Rather than wait for the

⁶ IEEE Std 802.3, 1998 Edition, “Part 3: Carrier sense multiple access with collision detection (CSMA-CD) access method and physical layer specifications, Copyright 1998, IEEE, ISBN 0-7381-0330-6.

corresponding BPDU information to time out, a link integrity failure may alternatively be detected by a link integrity test which operates at the physical layer of the protocol stack. The link integrity test typically exchanges test messages across the respective link 128 at a relatively high rate (e.g., every 10 milliseconds).” Gai, 12:4-9.

86. While it is not explicitly described in the STP standard, Gai, or many other references, it must be noted that each communications link must be attached to a switch via a port on the switch: each link is connected at each end to a port on the two switches it connects. For as many links as are connected to a switch, either upstream or downstream, there must be the same number of ports. The ports provide the physical connection to the links. For instance, in Fig. 1 of Gai, the ports of access switch 114 are explicitly labeled but similarly configured access switches 115 and 116 are not. Likewise, link 128 is shown connecting access switch 114 to backbone switch 122, but no ports are labeled on backbone switch 122, nor are they labeled on backbone switches 120, 121, 123, 124, or 125. This is not because these switches do not have ports but rather to simplify the figure. Gai provides further support for the existence of these ports, making it obvious to a POSA that all switches described follow the same structure. Gai, 7:34-37. In short, ports exist implicitly in every switch for every link in which the switch

participates, and at the very least it would be obvious to use such ports because that is the only practical way for messages to be received at a switch.

2. Link Failure Recovery

87. In the event of a link failure detected by the physical level, while there is no recovery mechanism provided at the physical layer, the occurrence of the failure is reported to higher level control functions in the bridge, which can begin restoration of the connection, generally through a different physical link connected to a different bridge port.

88. When a bridge detects a link failure, the bridge triggers a process to failover to a new root port, which is shown in Figures 19-20 below.

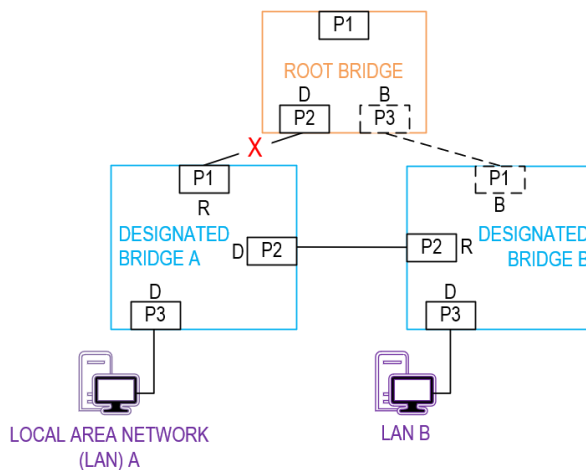


Figure 19 - STP link failure

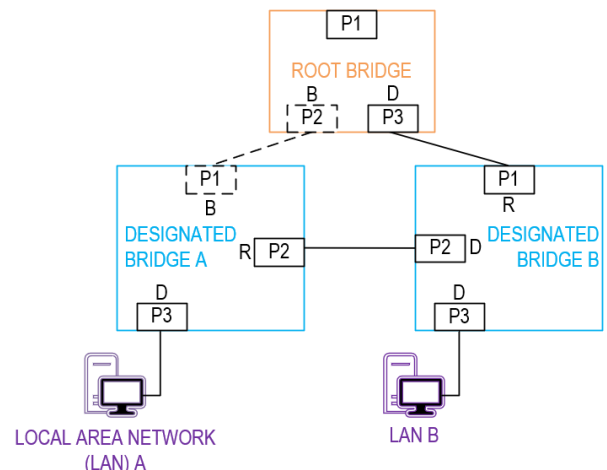


Figure 20 - STP link recovery

89. In Figure 19 above, designated bridge A detects a failure on its root port (identified by a red “X” through the link between that root port and the designated port on the root bridge upstream) because it does not timely receive a BPDU on its root port. As shown in Figure 20, responsive to the failure,

designated bridge A transitions its port P1, through intermediate states, into the blocking state, and marks its port P2 as a candidate new root port. Placing port P1 into the blocking state deactivates the previously-active link. The newly-deactivated link is shown in Figure 20 as a dashed line between the root bridge and designated bridge A. Port P2 of the root bridge may be placed in a blocked state, though as previously noted, another configuration would allow it to continue in a forwarding state since any messages it sends to port P1 of designated bridge A would be discarded once port P1 of designated bridge A has been blocked.

90. Designated bridge A transmits BPDUs via its port P2 to notify designated bridge B of the change. Designated bridge B transitions its port P1 from the blocking state, through intermediate states, to the forwarding state, activating the previously-inactive link between designated bridge B and the root bridge (solid line in Figure 20). Network operation then resumes, with designated bridge B receiving and forwarding messages on the newly-active link. The overall delay from designated bridge A receiving the last BPDU until reconfiguration is lengthy (approximately 50 seconds), frustrating timely communication.

D. Extensions to STP

91. Like any standards document, STP is a living document with ongoing work to address shortcomings that are discovered through use of the protocol or research of the many people investigating the technology. Two specific extensions

are described below – one developed by an IEEE standards body specifically to extend STP, the other by a group of researchers investigating the operation of IEEE 802 networks.

1. IEEE 802.1W-2001 Rapid Spanning Tree Protocol

92. The IEEE 802.1 working group that was addressing the standards needed to ensure orderly operation of data networks recognized the limitations of IEEE 802.1D-1998 namely, the long time needed to detect network failures or reconfiguration and to recover from these changes. For instance, with the standard BPDU retransmission interval and the standard number of lost messages before recovery would be possible, it might take up to 50 seconds to restore a connection after a network failure. For this reason, the IEEE 802.1W-2001 extension to STP was addressed. This extension is sometimes referred to as RSTP, the Rapid Spanning Tree Protocol. There are several key differences between STP and RSTP that will be described below.

93. The first difference between STP and RSTP involves the roles that bridge ports may play. While STP defines Root ports, Designated ports, and Blocked ports, RSTP eliminates the Blocked port role and replaces it with two new port roles: Alternate port and Backup port. Both the Alternate port and the Backup port are in the Blocking state that is, they are listening to traffic but not forwarding it. The Alternate port is similar to a Blocked port in that it is listening

to traffic from the Root bridge but is not processing that traffic. In essence, the Alternate port may be considered to be the second best Root port for providing a least cost path to the Root bridge using a different network segment serving as an alternate to the Root port, ready to take over if the Root port encounters a failure on its associated link. In contrast, the Backup port serves as an alternate to the Designated port, providing a potential replacement in the downstream direction (away from the Root bridge), ready to take over if the link from the Designated port fails.

94. RSTP adds another new type of port role to STP, an Edge port, which is a port that is not connected to other bridges but, rather, is connected to end devices, like computers or workstations. Since these devices are at the ends of the network, they cannot create loops, so the associated ports in Edge port mode are not involved in the RSTP. The Edge ports are in Forwarding state. The various port roles defined in RSTP are summarized in Table 3 below.

Port Role	Function
Root Port	The bridge port connected to a link with the least cost path to the Root Bridge
Designated Port	The bridge port connected to the link that is connected to a Root Port on the next bridge
Alternate Port	The bridge port is connected to a link that can provide an alternative second best path to the Root Bridge
Backup Port	The bridge port is connected to a link that is connected to an Alternate Port on the next bridge
Edge Port	The bridge port is connected to an end device, such as a workstation, not another bridge.

Table 3 - RSTP port roles

95. The second difference between STP and RSTP is in terms of the port states. Unlike the five states of STP (Forwarding, Learning, Listening, Blocking, and Disabled), RSTP simplifies the protocol with three states: Forwarding, Learning, and Discarding. Like the STP ports in the Blocking state, RSTP ports in Discarding state do not process or forward frames but do listen for BPDUs. The port states are summarized in Table 4 below.

Port State	Function
Forwarding	The port will process BPDU messages, receiving and forwarding frames
Learning (transitory)	The port learns about BPDUs
Discarding	The port listens to BPDUs but discards them
Blocking	The port receives BPDUs but drops (ignores) them

Table 4 - RSTP port states

96. The third difference between STP and RSTP is that RSTP can provide faster detection of changes in the network topology. In both STP and RSTP, there is a topology change notification (TCN BPDU) message to allow a bridge to announce a topology change it detects in the network. Where an STP bridge must send the TCN BPDU to the Root bridge which, in a second step, will distribute a topology change announcement (TC BPDU) message to the entire network, in contrast an RSTP bridge that detects a network topology change sends a TC BPDU message itself. The bridge detecting a network topology change is said to “flood”

the network with the announcement of a change, sending the TC BPDU message upstream and downstream, with each bridge that receives the message forwarding it upstream and downstream, in turn, to propagate the change everywhere. Each bridge along the way distributes the announcement up and down the tree to inform the other bridges in a one-step operation, propagating network topology change information through the network more quickly.

97. Because of the changes to the response to network failures and the streamlined response to network topology changes, RSTP provides a much faster update to network changes. As described above, STP would require up to 50 seconds to recover from a network failure:

- a. Twenty seconds after a port stops receiving Hello BPDU messages to act on the network failure (the Max Age time), plus
- b. Fifteen seconds in the Listening state (the Delay Forward time), plus
- c. Fifteen seconds in the Learning state (also the Delay Forward time)

98. In contrast, rather than waiting 50 seconds, RSTP can respond to network failures in 6 seconds. Instead of waiting the Max Age time before acting, with RSTP, the bridge only waits for three missing Hello BPDU messages, each repeated every two seconds. The port immediately switches through the Listening state without delay and, instead of using timers to change states, uses a proposal/agreement plus synchronization protocol.

99. In addition to the network recovery and reconfiguration differences noted above, RSTP also introduces the identification of “link types.” Point-to-point links operate in full duplex mode and benefit from the rapid transitions of RSTP while Shared link are links that operate in half duplex mode and do not participate in the rapid transition protocol.

2. IEEE 1999

100. The IEEE 802.1W-2001 RSTP standard was, by no means, the only extension to STP that has been considered. Varadajan and Chiueh (“IEEE 1999”)⁷ addresses the same challenges as Gai and the ’986 patent with respect to the IEEE 802.1D standard. EX1009, Abstract (IEEE 1999 discloses a “fast spanning tree reconfiguration algorithm...a significant improvement over the standard IEEE 802.1d implementation”), 161 (“IEEE 802.1d was written with best effort traffic in mind and has several drawbacks in a real time connection oriented environment.”),

⁷ S. Varadarajan and T. Chiueh, “Automatic fault detection and recovery in real time switched Ethernet networks,” *IEEE INFOCOM '99. Conference on Computer Communications. Proceedings. Eighteenth Annual Joint Conference of the IEEE Computer and Communications Societies. The Future is Now (Cat. No.99CH36320)*, New York, NY, USA, 1999, pp. 161-169 vol.1, doi: 10.1109/INFCOM.1999.749264.

162 (“these timers are conservatively large, resulting in a large fault recovery time...Because data can only be transmitted through links belonging to the new spanning tree, all preexisting connections through the network have to be torn down.”), 164 (“While the IEEE 802.1d protocol works well for best effort traffic on transparent switched Ethernet networks, it has several shortcomings in a real time connection oriented environment...Coupled with a large spanning tree reconfiguration (fault recovery) time, network faults can result in large interruptions in service in IEEE 802.1d based networks.”), 165 (the “fault recovery mechanism has to be fast, so that the affected connection can be reestablished with minimal interruption in service”). IEEE 1999 extended STP and recognized the value of sending signaling to upstream and downstream nodes, increasing the speed of recovery, and maintaining locality of alerts to minimize overhead.

101. In the introduction, IEEE 1999 states “the fault recovery mechanism should constrain the effect of the fault to the affected link(s)/switch(es) and the connections through them, while allowing normal operation of the rest of the network.” This is the idea of maintaining locality of reconfiguration and reducing the radius of the changes to the spanning tree to accelerate recovery noted above.

102. Further, IEEE 1999 recognizes that failure information preferably is conveyed both upstream and downstream whenever a failure is detected. For example, in ¶3.4 – Fault Recovery, it is noted that “In the interior of the network,

the fault would be independently detected by switches both upstream and downstream of the fault.” This is true because, since part of the detection method proposed by this paper is to use the link health monitoring and a link failure will be detectable at both ends of a link when the 10 millisecond interval “heartbeat” messages are no longer being received.

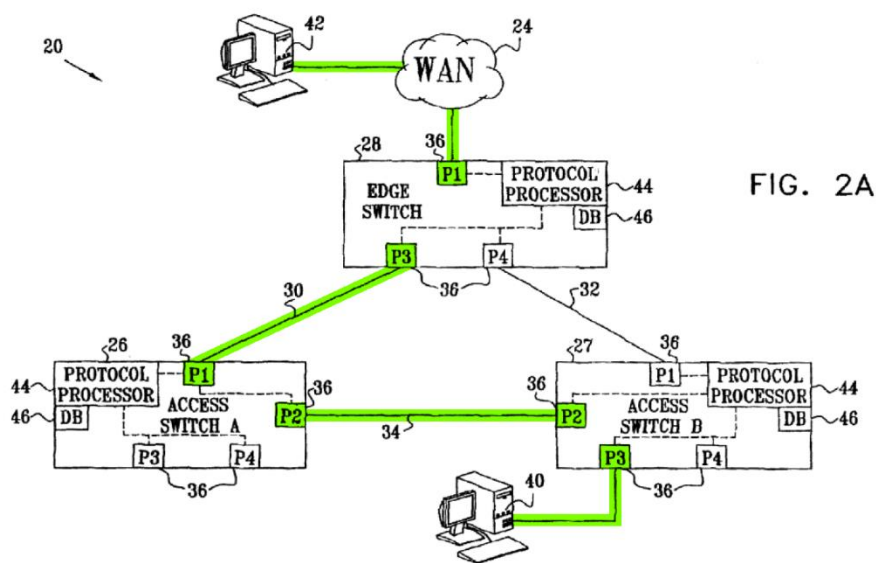
IV. THE '986 PATENT

103. The '986 patent purports to provide “rapid reconfiguration in response to network topology changes” (EX1001, 3:58-60) in Spanning Tree networks. EX1001, 8:1-9. In operation, a link is placed in an active state via connection to an active port of a network switch. EX1001, 8:10-13. An active port is placed in a forwarding state in which the port receives and transmits data frames. EX1001, 2:39-41. A link that is redundant to the active link is connected to a port in a blocked state until the active link connected to the active port fails. EX1001, 9:26-28. Following link failure, the '986 patent purports to reconfigure a network more quickly than standard STP self-healing by immediately activating a back-up port and using a dummy frame mechanism. EX1001, 3:1-14, 4:13-31, 10:16-24.

104. The '986 patent claims use the phrase “dummy traffic.” The '986 patent specification uses the phrase “dummy message” as a synonym for “dummy traffic.” EX1001, 3:29-30 (“The switch then transmits dummy multicast messages toward the root...”). Claims 6, 11, 18 and 23 state that “dummy traffic comprises

data frames.” Gai discloses “dummy multicast messages” (EX1005, 5:54-60) and also discloses that messages and frames are essentially synonymous. EX1005, 7:31-34. Accordingly, the dummy messages of Gai are synonymous with the dummy messages discussed in the ’986 patent specification and the dummy traffic claimed in the ’986 patent claims.

105. Returning to data frame forwarding as conveyed by the ’986 patent, the network path for data frame forwarding prior to link failure is highlighted in green in an annotated version of Figure 2A, below.



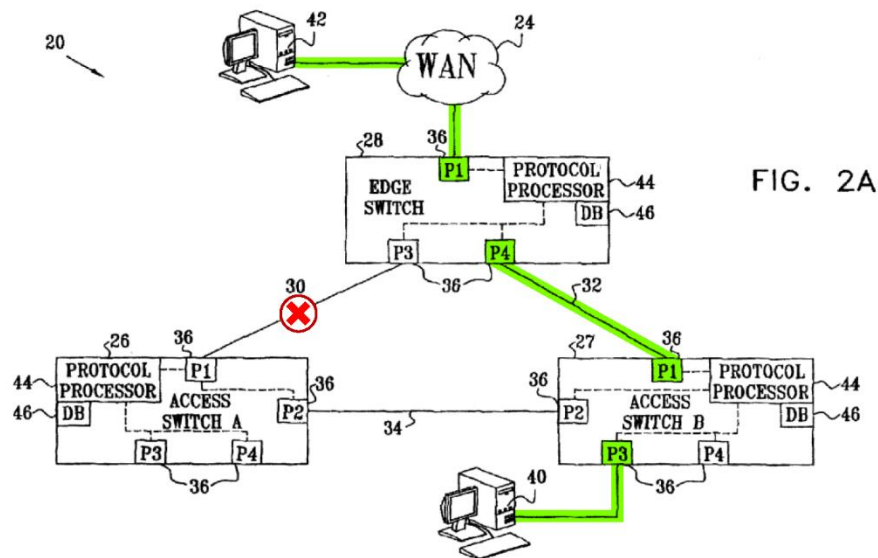
Annotated Figure 2A of the ’986 patent before link failure

106. When a failure is detected in an active link (e.g., link 30), one of the blocked ports (e.g., port P4) of the edge switch is chosen to serve as the new active link and the failed port (e.g., port P3) is blocked. EX1001, 9:26-35. So-called dummy frames are sent from the edge switch through the new active link (e.g., link

32) to downstream access switches (*e.g.*, access switch 27). EX1001, 9:42-44.

Upon receiving these dummy frames, the receiving access switches (*e.g.*, access switch 27) use the learning algorithm specified in the IEEE 802.1D standard to rebuild their filtering databases, which are used to forward data frames, so that upstream traffic will be directed to the new active link. EX1001, 9:62-10:4.

107. Responsive to the detection of the failed link, the updated network path for data frame forwarding is highlighted in green in the annotated version of Figure 2A below. The previously-active link between the edge switch and the downstream access switch A, now blocked after detecting the failure, is identified by a red X. The edge switch and the downstream access switch B exchange traffic over the newly-active link (*e.g.*, link 32).



Annotated Figure 2A of the '986 patent after link failure

108. Because various time-consuming aspects of STP are not utilized, the self-healing process of the '986 patent is not subject to the 50-second delay of STP. By using a dummy frame routing update, the '986 patent asserts that its self-healing process “can be completed in well under a second, after which normal operation of the network can resume.” EX1001, 10:22-24.

A. The Challenged Claims

109. Independent claim 1 of the '986 patent is reproduced below with labeling added for reference.

1[PRE] A method for network communication, comprising:

- [1A]** coupling an edge switch in a network to one or more access switches via at least first and second redundant links, respectively connected to first and second ports of the edge switch;
- [1B]** placing the first port in an active state, while placing the second port in a blocking state;
- [1C]** conveying communication traffic over the first link while the first port is in the active state, thereby causing each of the switches to build a respective database for use in forwarding the traffic;
- [1D]** responsive to a failure associated with the first link, placing the second port in the active state and the first port in the blocking state; and
- [1E]** sending dummy traffic from the edge switch over the second link to the one or more access switches, so as to

cause each of the one or more access switches to modify its respective database responsive to the second port being in the active state.

B. Prosecution History

110. The challenged claims were filed on January 7, 2002. EX1002, 1. The Patent Examiner issued no Office Action on the merits. The challenged claims were granted in U.S. Patent No. 6,917,986 on July 12, 2005. IEEE 802.1D, published in 1998 (EX1007; EX1008, 5), was of-record. EX1001, 1.

C. Level of Ordinary Skill in the Art

111. A POSA with respect to the '986 patent would have a bachelor's degree in electrical engineering or a related field, plus at least two years of industry experience or graduate-level academic experience with computer networking.

112. Prior to the date of the '986 patent, I had at least the level of skill in the art of a POSA. In addition, through my career in the various positions I have held, I have supervised and interacted with numerous people with this level of skill in the art. I have subsequently taught students with this level of skill of the art.

V. UNPATENTABILITY GROUNDS

A. Ground 1: Claims 1-3, 6-15, and 18-24: Obvious Over Gai in View of IEEE 1999

1. U.S. Patent No. 6,032,194 (“Gai”)

113. Gai, like the '986 patent, is directed to rapidly reconfiguring a Spanning Tree network. EX1005, 5:35-39. Gai recognizes several disadvantages of the STP algorithm, including that, during the time-consuming reconfiguration process, message delivery is delayed, which network users may perceive as a service disruption, presenting significant problems on networks intended to be highly reliable. EX1005, 4:66-5:7. Additionally, following the reconfiguration process, BPDUs may flood the network, thereby consuming substantial network resources. EX1005, 5:7-13. Gai addresses these issues by immediately transitioning port status and utilizing a dummy packet routing update to quickly inform the network of newly-activated ports, without the delays related to STP's use of BPDUs and STP's requirement that ports transition through multiple states. EX1005, 6:1-5, 14:54-63.

a. Port Functions And Switch Connectivity

114. All routers, bridges and switches have a variety of ports to connect them to other devices. This would be obvious to a POSA when applying the teachings of the STP standard, Gai, or the '986 patent. As described above, when a device is using the STP, RSTP standard, or Gai, the ports will operate in various

states and will have various roles at different times, similar to the descriptions outlined above. In particular, during normal operation, a port will receive and forward traffic along the communications link to the port on the next device if it is in a forwarding state and will otherwise listen to traffic on the link but will not act on the traffic if it is in a blocked state. All this will occur under the control of the software in the switch.

115. Traffic received on an active port, that is, one that is in a forwarding state, will be examined within the switch to determine how it should be processed. Source and destination addresses contained in the traffic will be compared to the information in the forwarding database and, based on the fields in the database, traffic may be forwarded to a specific output port on the switch so it may be sent along the corresponding link to the next switch or destination device. Again, it needs to be understood that the operation of a switch is the same, whether it is one of Gai's access or backbone switches, one of the '986 patent edge switches, or a switch operating under the control of the STP or RSTP standard.

116. It is common in large, distributed networks to create sub-networks that use different addresses than the other sections of the network. This may be done for security reasons or to minimize traffic on the wider network. As their name implies, the filtering databases that are present in all the switches described in Gai perform a filtering function. That is, they may filter some of the traffic

arriving at a switch before forwarding it to other switches on the network. A POSA would understand that traffic arriving at a switch in Gai's network with a particular source address could have that address translated to a new address using information copied from the filtering database.

117. Gai's protocol entity 232 includes a spanning tree engine 236 that processes traffic in accordance with the STP and relies on the spanning tree table 242 and filtering database 244 to process traffic and respond to network changes and failures.

118. The forwarding entity 238 uses the filtering database 244 to instruct the switch 214 how traffic should be forwarded. The filtering database has a number of entries each containing a destination address, the port which traffic destined for the destination address should be forwarded to and a timer field. Since the failure of a switch or a link will stop the flow of traffic, it is necessary to ensure the timeliness of routing information. Having a timer associated with the ports and destination addresses processed by the switch allows the switch to detect "stale" information in the filtering database to begin the recovery process.

b. Port Reconfiguration After Link Failure

119. Under normal startup conditions, Gai finds a minimum spanning tree in the same manner the STP standard describes. As described elsewhere, Gai provides an additional set of commands to be executed by the network

administrator at switches that are in a position to best serve as a Root Bridge and a secondary Root Bridge. In addition, the network administrator can execute commands at certain switches to ensure they can never be the Root Bridge.

120. During normal network operation, the network's spanning tree will not change and traffic will be routed using the filtering database to route traffic to its destination. However, if a switch fails somewhere in the network, if one of the links between switches fails, or if there is switch or link that is added to or removed from the network causing a reconfiguration of the network, the original spanning tree will need to be changed. A failure or removal of a switch or link will eliminate some connectivity from the network, causing the network to split into two or more disjoint segments. An additional switch or link may create a loop in the network with the attendant susceptibility to the "broadcast storm" described elsewhere. For these reasons, the spanning tree will need to be modified and it is Gai's intent that this be performed as quickly as possible.

121. Part of Gai's rapid recovery strategy is the manual designation of a backup Root Bridge, as described above. The other is identification of a backup root port using the rapid reconfiguration entity 234. When the spanning tree is first created by the network, besides the root port with the lowest network cost being selected, one of the blocked ports is identified which will have the second lowest network cost. While this backup port is normally not used when the desired root

port is available and forwarding traffic, in the event of a failure or reconfiguration making that port unavailable, the backup port can quickly be selected to take over.

122. After detecting a link failure, Gai and the '986 patent both reconfigure their ports in the same manner. EX1001, 3:26-35 (describing Gai's operation); 4:14-30 (describing the '986 patent's operation). In Gai, a downstream switch detects failure of an active link 128 at an active port (i.e., port 3), which is illustrated below in an annotated version of Gai's Figure 1.

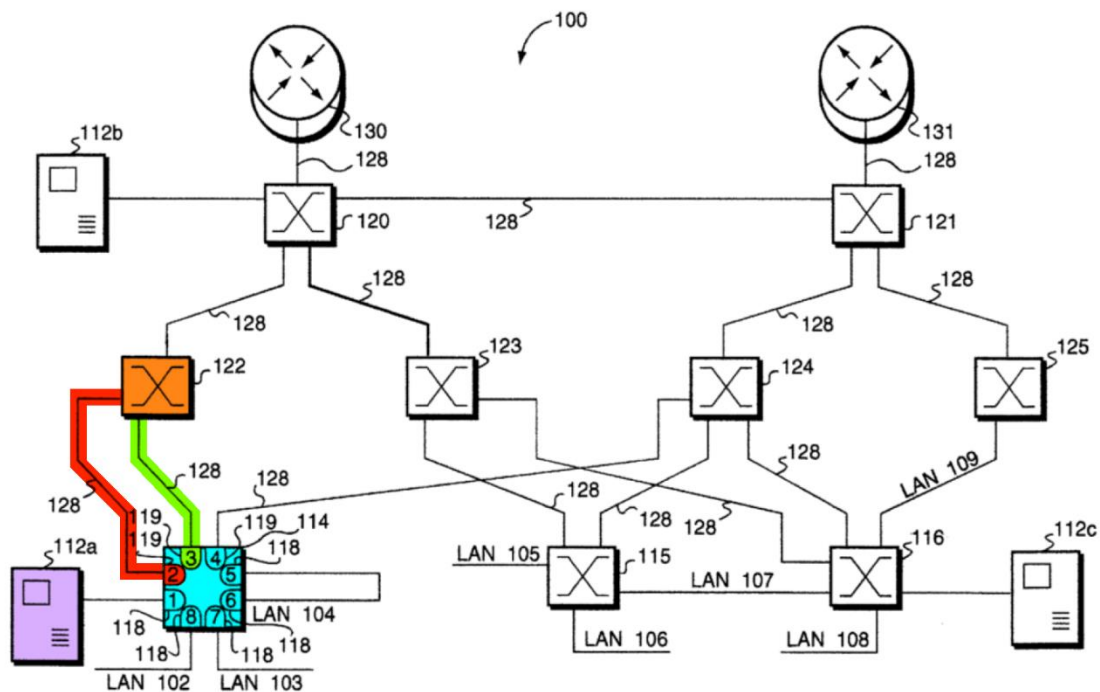


FIG. 1

Annotated Figure 1 of Gai before link failure

123. Below, another annotated version of Figure 1 shows port reconfigurations of the downstream switch after link 128 fails. After detecting the

failure, the downstream switch places the previously-active port in a blocking state, changing previously-active link 128 to an inactive link. EX1005, 14:40-51. The previously-blocked port is placed in an active state, changing the previously-inactive link 128 to an active link. EX1005, 11:48-52. Like the '986 patent, by immediately transitioning the ports' states without progressing through intermediary states (e.g., listening and learning STP states), Gai reduces the time needed to effectuate the network topology change. EX1005, 5:51-53, 6:50-53.

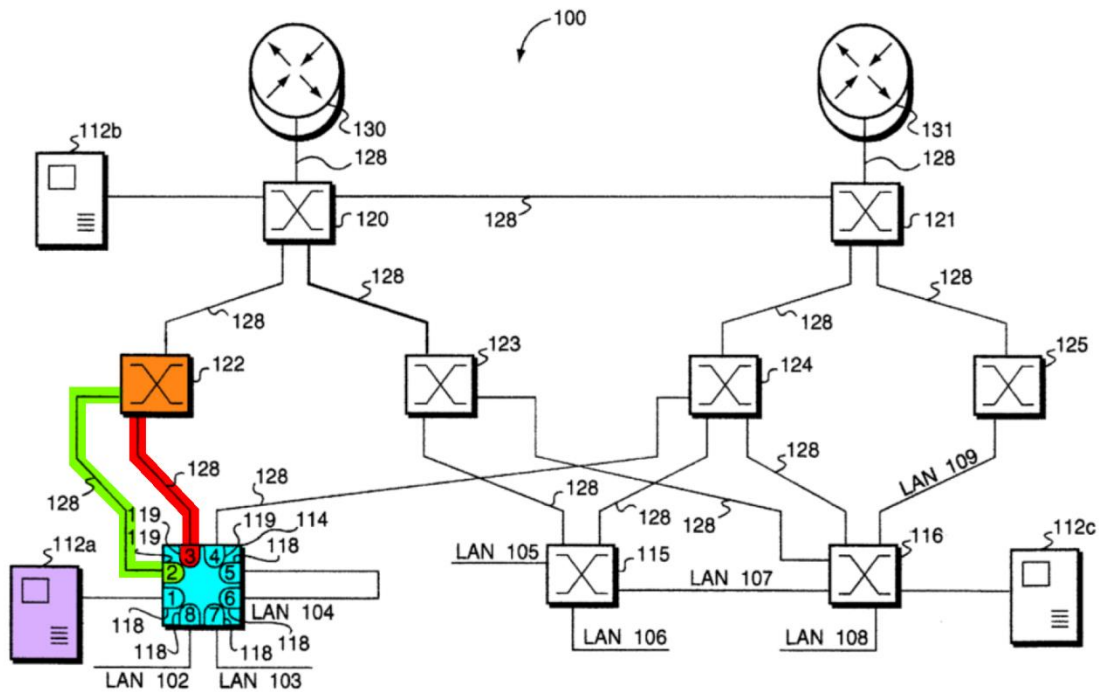


FIG. 1

Annotated Figure 1 of Gai after link failure

c. Dummy Traffic

124. Gai defines “dummy multicast messages” and illustrates them in Figure 4. These messages contain source and destination addresses (SA and DA as noted in the figure), which convey a multicast destination address to cause the messages to be distributed through the network as well as the source address of the device originating them. By including the source address, devices receiving the multicast address can identify devices that are reachable via the path the multicast message was received from. By convention, IP networks, multicast addresses have specific default ranges from 224.0.0.0 through 239.255.255.255. The dummy messages defined by Gai serve the same purpose as the dummy frames described by the ‘986 patent, notifying other switches of port configuration changes. EX1001, 3:29-35 (describing Gai’s operation), 4:17-28 (describing the ‘986 patent’s operation).

125. To notify upstream switches, Gai’s downstream switch transmits dummy traffic upstream through the new active link to update the filtering database of the upstream switch. EX1005, 12:56-66. In summary, Gai discloses setting up a spanning tree network, performing port reconfiguration after a link failure, and transmitting dummy traffic, all as described in the ‘986 patent.

d. Network Initialization Commands

126. One of the functions described in Gai is the use of novel commands that are not present in the STP standard. STP uses a series of protocol exchanges as described above to allow a series of nodes to negotiate to find the Root Bridge. To give the network administrator more flexibility in configuring the network, Gai defines the `Become_Root_Primary` command (EX1005, 6:13) to change the Bridge ID to a value that will guarantee it to become the Root Bridge, independent of its position in the network and networking costs. As an example, since STP uses 16-bit values for BID, values ranging from 0 to 65535 are possible values for BID. The default STP BID is normally in the middle of the range, that is 32768. The `Become_Root_Primary` sets the BID for the bridge that is to become the Root Bridge to a value of one quarter of this or 8192. Gai also defines a command, `Become_Root_Secondary`, to designate a second bridge that can be used as the Root Bridge, should the Root Bridge fail. This command sets the desired secondary bridges BID to an intermediary value of 16384.

127. The novel commands described by Gai are optional and would only be used when the network administrator configures the network initially. Once the network is set up and the BIDs set to the values that will ensure the primary and backup Root Bridges are designated, the commands need not be use again unless there is some major, probably manual, restructuring of the network.

128. Without Gai's novel commands, STP would operate as it would normally as I have described in the diagrams above. If the default STP value of 32768 for all the bridges' BID were used, the bridge MAC address and network costs would cause one bridge to be identified as the Root Bridge, as STP normally does. Gai provides a simple mechanism for the network administrator to use their knowledge of the interconnected networks to optimize the connection, if desired. For instance, it would be logical to designate a specific bridge as the preferred Root Bridge if that bridge had some particular network connections, for instance, to an outside network or a specific server location, but not doing this would not prevent STP from operating. Similarly, the network administrator might realize that it would be best to have a specific bridge as the Root Bridge if the first choice for the Root Bridge failed, but again, STP would find a replacement for a failed Root Bridge on its own without direct guidance.

129. Gai also identifies so-called "access switches" and provides a novel `Enable_Uplinkfast` command to initialize their BID. Again, assuming the default STP BID value being 32768, Gai describes the `Enable_Uplinkfast` as setting the BID value to a higher value, specifically 49152. In this way, Gai guarantees that a network administrator executing the `Enable_Uplinkfast` command at the access switch will guarantee that it can never be a Root Bridge of the network since all other bridges, other than other access switches, will have higher bridge priorities

when it comes to assigning the Root Bridge status. As it was the case for the Become_Root_Primary and Become_Root_Secondary commands, this command is only executed when the network is first established and is optional and only intended to optimize the overall operation of the network. An access switch is not appropriate to be assigned the function of a Root Bridge, given its position in the network, so Gai's method simply allows STP to initialize faster, but STP would work without it, just not as optimally. Since a network may have a number of access switches, it is reasonable to expect that a network administrator may choose to execute the Enable_Uplinkfast command at all of the access switches, preventing all of them from ever becoming Root Bridges.

130. The '986 patent purports to differentiate itself from Gai by not requiring, as part of the network self-healing process, certain network initialization commands discussed in Gai. EX1001, 3:35-39, 4:36-43. However, as discussed above, POSAs would recognize that Gai's self-healing technique does not require these commands, and that Gai discloses them merely as an example implementation for pre-configuring a network into an optimized spanning tree structure to streamline self-healing in the event of a link failure by taking into consideration server loads in a network, something not addressed in the STP algorithm. EX1005, 8:59-65, 10:29-36.

131. Even setting aside this optimization teaching, the '986 patent's claims do not exclude such commands in any event. I see nothing in any of the claims that precludes such commands.

2. IEEE 1999

132. The paper "Automatic Fault Detection and Recovery in Real Time Switched Ethernet Networks" by Srinidhi Varadarajan and Tzi-cker Chiueh was presented at the IEEE INFOCOM '99 Conference on Computer Communications (hereafter referred to as IEEE 1999). Although I did not attend this particular session of the conference, I have attended other sessions of this conference series and have attended many other IEEE conferences. It is routine for the conference organizers to distribute copies of all the papers presented at the conference in the form of several printed volumes or, more recently, as a CD-ROM so all the attendee and other interested people can access the materials presented. Further, the INFOCOM conference is a well-known conference and POSAs know (and knew before the '986 patent) that proceedings from the INFOCOM conference relate to the technology area covered by the '986 patent. Even before the advent of the IEEE online database, POSAs routinely reviewed the proceedings of this conference for relevant technical information in the field of the '986 patent.

133. IEEE 1999 addresses the same issues as the IEEE STP standard as the paper title suggests – it is intended to deal with fault detection and recovery

automatically. Specifically, this paper discusses several concepts that a POSA would recognize add desirable improvements to the STP standard. Like RSTP, IEEE 1999 recognizes the usefulness of having alternate paths through the network to take over if a link or switch fails, breaking one portion of the network tree off from the rest. IEEE 1999 does this without creating the loops that redundant active links would. Next, the authors recognize that, beyond the slower network signaling at the MAC layer and above used in STP, they can utilize faster link-layer failure detection methods. Additionally, IEEE 1999 recognizes the usefulness of localizing network failure detection and restoration, providing faster restoration and minimizing the need for widespread network involvement. Finally, this paper explicitly includes upstream and downstream signaling of failures, allowing bridges closer to the Root bridge as well as those further from the Root bridge to be aware of a network failure and begin to respond to it.

134. IEEE 1999 explains that switches both upstream and downstream of a link failure detect the failure. EX1009, 166 (“the fault would be independently detected by switches both upstream as well as downstream of the fault”). Each switch that detects a failure takes steps to rebuild the spanning tree. EX1009, 166 (“The fault recovery mechanism on each switch that detects the fault...[r]ebuild[s] the spanning tree”). As part of the spanning tree reconfiguration, and to improve recovery time, each switch that detected a link failure sends “teardown messages”

to both upstream and downstream switches to “terminate the connections through a failed link.” EX1009, 167.

135. POSAs would have understood that a switch that receives a “teardown message” updates its filtering database such that the failed connection is terminated. Thus, IEEE 1999 discloses to POSAs that, to improve recovery time, a switch, responsive to detecting a failure upstream or downstream of itself, sends messages both upstream and downstream to cause other switches to update their respective filtering databases. One salient feature of IEEE 1999 that is intended to improve network reliability and decrease failure recovery time is stated in the introduction: “we present our software based fault detection and recovery mechanism that delivers fast spanning tree reconfiguration and the ability to send data through links that do not belong to the spanning tree, thus localizing the effect of network faults.” EX1009, 162. They recognize the danger of redundant links creating the potential for loops and, in ¶3.0 – Fault Handling, state: “[b]roadcast loops are avoided by superimposing a logical spanning tree upon the physical topology to enforce a single route between any pair of hosts. Links that do not belong to the spanning tree are used to create a new spanning tree in the case of failure of links in the current spanning tree.” *Id.* Finally, the authors of IEEE 1999 make it clear that they intend to keep all links active to quickly detect and recover from failures, specifically in ¶3.3 – Fault Detection: “[s]witches in the network

send periodic (100 ms in the current implementation) inter switch hello messages on ***all*** their ports.” EX1009, 166 (emphasis added).

3. Gai in View of IEEE 1999

136. The '986 patent admits Gai discloses sending dummy traffic upstream from a switch that detects a link failure on its root port. EX1001, 3:26-33. Gai discloses sending dummy traffic to reduce link failure recovery time. EX1005, 6:42-53, 15:21-25. Aside from the network initialization commands discussed above (which, as noted above, are not excluded by the '986 patent's claims), the '986 patent purports to distinguish Gai by sending dummy traffic downstream, rather than upstream as in Gai. EX1001, 4:35-43.

137. In view of IEEE 1999, POSAs would have had reason (i.e., faster recovery from a link failure as explained in IEEE 1999) to modify Gai's technique by sending Gai's dummy traffic downstream after a downstream link failure (as the '986 patent describes), in addition to sending dummy traffic upstream after an upstream link failure (as Gai describes). This has the same benefit already described in Gai, *i.e.*, reducing the time needed for network self-healing. EX1005, 6:42-53, 15:21-25.

138. As noted above, IEEE 1999 discloses sending such messages both upstream and downstream, after detecting a failure either upstream or downstream, for the same purpose that Gai describes sending dummy traffic: increasing the

speed of self-healing following link failure in an 802.1D network. In view of IEEE 1999, POSAs had reason to use Gai's technique when a failure is detected downstream of a switch, just as when a failure is detected upstream of a switch. While IEEE 1999 describes the messages as "teardown messages," their function is the same as Gai's "dummy messages."

139. Gai discusses two types of switches: access switches and backbone switches. "Access switches are preferably coupled to entities (e.g., LANs, end stations, etc.) whereas backbone switches provide the interconnections between access switches." EX1005, 6:9-11. Gai discloses entering an Enable_Uplinkfast command at an access switch (e.g., access switch 114) to block a failed port (e.g., the root port) and configure the access switch to send dummy traffic upstream through a new active port. EX1005, 6:39-46. When modified in view of IEEE 1999's disclosure that self-healing network traffic can advantageously be sent both upstream and downstream, the command would be entered at a backbone switch (e.g., backbone switch 122) to block a failed port (e.g., a designated port connected to access switch 114) and configure the backbone switch to send dummy traffic downstream to an access switch (e.g. access switch 114) through a new active port. While Gai uses its protocol signaling command, Enable_Uplinkfast, to send dummy traffic through a new root port, POSAs would deem it an obvious implementation detail to modify the parameters of the command to be applicable to

other ports, such as a designated port on a backbone switch. For instance, POSAs could modify the parameters of the Enable_Uplinkfast command to affect operation of a designated port such that the designated port can send dummy traffic through the designated port upon detection of a failure.

140. Claim 1 of the '986 patent requires an "edge switch" to send dummy traffic downstream to an access switch. Gai's backbone switch 122 is an "edge switch." Per the '986 patent, "[c]ore switches interconnect the access switches into a Wide Area Network (WAN) by direct connection between the core switch and the access switches and/or by connection to other core switches." EX1001, 1:33-36. Gai's backbone switch 122 is a "core switch," as it connects directly to access switch 114 and, via another backbone switch 120, to router 130, which POSAs would have understood connects to a WAN, or at least it would have been obvious to connect to a WAN. EX1005, Fig. 1. As Figure 1 shows, router 130 and router 131 only connect to backbone switches 120 and 121, respectively, with no other connection indicated. Obviously, as shown, these routers 130, 131 would serve no function in the network, but POSAs would understand that this is only because Gai's illustration has focused on the details of the '194 patent's invention and not the broader context that it can be used with. Thus, POSAs would have understood the intent of including the routers 130, 131 was to show that Gai's network can connect to other networks, including a WAN. EX1005, Fig. 1.

141. The '986 patent further explains: “Core switches that are directly connected to access switches are referred to herein as edge switches.” EX1001, 1:36-38. Because Gai’s backbone switch 122 connects directly to access switch 114, it is an “edge switch.”

142. Further, the '986 patent admits that Gai’s backbone switches (which the '986 patent calls “trunk switches”) are “comparable to the core and edge switches” of the '986 patent. EX1001, 3:21-24. In short, the modified Gai technique involves an “edge switch” sending dummy traffic downstream to an “access switch” as claim 1 requires.

143. Further, some claims of the '986 patent (*e.g.*, claim 12) use the term “bridge” instead of “edge switch” and “access switch.” “Bridge” is a general term encompassing both edge and access switches. EX1001, 1:26-38 (“[t]he most common type of MAC bridge is a switch...Switches can be roughly classified as access switches or core switches...”). Thus, for the same reasons described above, the modified Gai technique involves a “bridge” that sends dummy traffic downstream to another “bridge” as required, for example, by claim 12 of the '986 patent.

144. In short, when modified in view of IEEE 1999’s disclosure that it was advantageous to send self-healing messages both upstream and downstream after a link failure either upstream or downstream, Gai’s backbone switches (*i.e.*, the

claimed “edge switches” and “bridges”) send Gai’s dummy traffic both upstream and downstream, regardless of whether a link failure is detected upstream of the switch or downstream of the switch. This means that modified Gai’s backbone switch / edge switch / bridge sends dummy traffic downstream to Gai’s access switch / bridge, as recited in the challenged claims. Doing so achieves Gai’s goal (shared by IEEE 1999) of reducing the time taken to rectify network failures, including between an edge switch and an access switch. EX1005, 11:53-55, 11:64-12:12.

145. Finally, a POSA would have had a reasonable expectation of success in modifying Gai to have its backbone switches / edge switches / bridges send dummy traffic both upstream and downstream because signaling protocols are routinely modified to provide for different control signaling, such that there would have been no additional technical challenge in sending those messages both upstream and downstream. As described above, the STP standard and, since STP is incorporated into the ’194 patent by reference, Gai’s network is built on the IEEE 802.3 physical layer. This means that each of the ports on the backbone switches, edge switches, and bridges uses the signaling of IEEE 802.3. There is nothing preventing a POSA from using the upstream or downstream ports from sending dummy traffic and it would be logical for them to consider doing so if it added additional functionality to the system notably, alerting both upstream and

downstream bridges of locally detected failures. For example, Gai discloses that rapid reconfiguration entity 234 can be at any switch such as access switch 114 or backbone switch 122. EX1005, 7:62-64, 8:67-9:1. Therefore, there is no additional technical challenge to modifying backbone switch 122 because it already includes the component Gai discloses to generate and transmit dummy multicast messages. Since Gai's modified technique applies to backbone switches, which qualify as the claimed "edge switches" and "bridges," the combination of Gai with IEEE 1999 renders obvious the '986 patent's claims as explained below.

146. I reviewed, and had input into, the Petition that this Declaration accompanies. I endorse the discussions in the Petition, some of which (particularly in the claim analysis section below, but also in various places above) closely match discussions in my declaration.

4. Claim 1

a. [1PRE] A method for network communication, comprising:

147. Gai discloses "a method...for rapidly configuring a computer network following a network change." EX1005, 1:15-17. Gai discloses "a method...for balancing message traffic across several links of a computer network." EX1005, 5:29-30.

b. [1A] coupling an edge switch in a network to one or more access switches via at least first and second

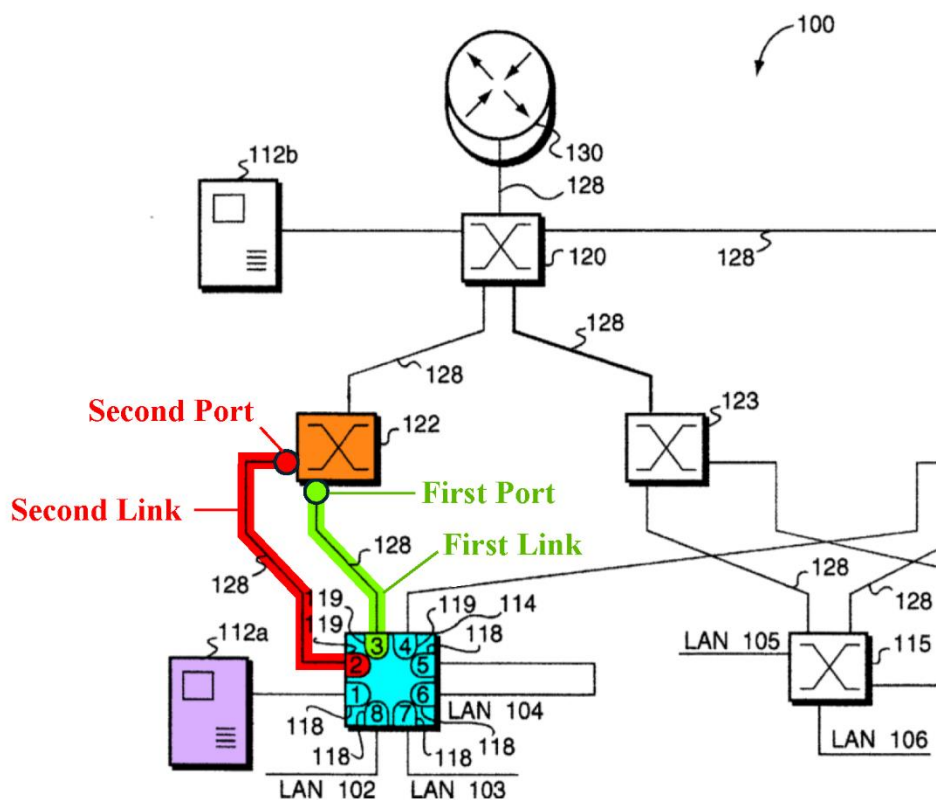
redundant links, respectively connected to first and second ports of the edge switch;

148. Gai discloses a computer network including backbone switches 120-125 and access switches 114-116. EX1005, 7:3-9; *see also* EX1001, 3:15-21 (acknowledging that Gai discloses “access switches”). The backbone switches 120-125 and the access switches 114-116 are interconnected through redundant communication paths that “prevent portions of the network 100 from becoming isolated should any constituent link or device fail.” EX1005, 7:53-57. Each of the backbone switches 120-125 and access switches 114-116 have multiple ports. EX1005, 7:34-37 (“[E]ach switch and router, such as access switch 114, identifies its local and trunk ports 118 and 119 by numbers (e.g., port number one, port number two, port number three, etc.).”), Fig. 1.

149. The '986 patent explains: (i) “[c]ore switches interconnect the access switches into a Wide Area Network (WAN), by direct connection between the core switch and access switches and/or by connection to other core switches” and (ii) “[c]ore switches that are directly connected to access switches are referred to herein as edge switches.” EX1001, 1:33-38.

150. As shown in annotated Fig. 1 of Gai below, backbone switch 122 is an edge switch because it meets both (i) and (ii) above. Backbone switch 122 meets (i), and thus is a “core switch,” because it interconnects access switch 114 into a Wide Area Network (WAN) by direct connection between backbone switch 122

and access switch 114 and via backbone switch 120 and router 130. EX1005, 7:26-30, Fig. 1, 6:10-12. As noted above, it would be obvious to a POSA that backbone switch 122 interconnects access switch 114 into a WAN through router 130. Backbone switch 122 meets (ii), and thus is an “edge switch” as claimed, because it directly connects to access switch 114.



Annotated Figure 1 of Gai before link failure

151. Gai thus describes a network topology that meets [1A]. Although not shown in Gai’s Figure 1 above, it would nevertheless be obvious to a POSA that backbone switch 122 has a plurality of ports for coupling to another switch, such as access switch 114. Thus, Gai implicitly discloses that backbone switch 122 has

ports that are respectively coupled to ports 2 and 3 of access switch 114. As shown in the annotated version of Gai's Figure 1 above, backbone switch 122 (*i.e.*, a claimed "edge switch") is coupled to access switch 114 via at least first and second redundant communication links 128, the "first" of which is an active link while the "second" of which is an inactive link. EX1005, 11:41-52. The active link is between port 3 of access switch 114 and a "first" port of backbone switch 122 (green circle). The inactive link is between port 2 of access switch 114 and a "second" port of backbone switch 122 (red circle). These ports on backbone switch 122 are disclosed and, at the very least, obvious.

152. In addition, the '986 patent specification effectively admits that Gai meets [1A]. EX1001, 3:18-24. The '986 patent specification describes that Gai's network has both core switches and access switches; some of which are called "trunk switches." The '986 patent admits "[t]hese trunk switches are comparable to the core switches and edge switches defined above." EX1001, 3:18-24. The '986 patent also describes that the Gai "manage[s] redundancies in their upstream ports" and that Gai's access and core switches have "multiple trunk ports providing connecting to the STP root." EX1001, 4:40-42; 3:18-21. Accordingly, the '986 patent effectively admits that Gai discloses switches, such as edge and access switches, that are coupled together through redundant links.

c. [1B] placing the first port in an active state, while placing the second port in a blocking state;

153. Gai explains that its switches place their ports in either a forwarding or blocking state. “The network preferably includes a plurality of devices executing the spanning tree algorithm so as to elect a root and place ports of the devices in either a forwarding or blocked state...one trunk port is in the forwarding state and the other trunk ports are in the blocked state.” EX1005, 5:36-43. With reference to the annotated version of Gai’s Figure 1 above, “port number three, which is coupled to backbone switch 122, is the root port for switch 114 and thus in the forwarding state...and port numbers two and four, which also connect switch 114 to the backbone switches (*i.e.*, backbone switches 122 and 124), are in the blocked state.” EX1005, 11:42-48.

154. Access switch 114 and backbone switch 122 are both switches that operate in accordance with STP and IEEE 802.1D. Therefore, like access switch 114 having ports that can be placed in either the forwarding state or the blocking state, backbone switch 122 can likewise place its ports in either the forwarding state or the blocking state. The corresponding ports on backbone switch 122 are placed in corresponding states, or at least it would have been obvious to do so for reasons I have already explained but will briefly reiterate. When a port on backbone switch 122 is communicating over an active link with a port on access switch 114, both ports must be in the forwarding state. When the link between

such two ports is inactive because the relevant port on access switch 114 is blocked, it would be obvious to block the corresponding port on backbone switch 122 because doing so would avoid the waste of having messages prepared and sent over such link that would only be discarded upon arrival anyway. By placing a port (*i.e.*, “first” port) of backbone switch 122 that is connected to port 3 of access switch 114 in the active state while placing a port (*i.e.*, “second” port) of backbone switch 122 that is connected to port 2 of access switch 114 in the blocking state, Gai meets [1B].

155. Further, the ’986 patent admits that Gai meets [1B]. EX1001, 3:24-26 (“One of the trunk ports of each switch is placed in the forwarding state, while the others are designated as back-up ports.”).

d. [1C] conveying communication traffic over the first link while the first port is in the active state, thereby causing each of the switches to build a respective database for use in forwarding the traffic;

156. Gai conveys communication traffic over the first link identified above (between port 3 on access switch 114 and a corresponding port on backbone switch 122) while those ports are in the active state. EX1005, 7:30-33.

157. As discussed above, databases for forwarding traffic are known from conventional STP. EX1007; EX1005, 1:50-58. Gai discloses such databases as shown in a pink box in an annotated version of Gai’s Figure 2 below. Figure 2 is a partial block diagram of a switch 214 (which “corresponds to access switch 114 of

Fig. 1,” EX1005, 7:62-64), including filtering database 244, “configured to store address information corresponding to the entities of network 100.” EX1005, 7:62-64; 8:31-33.

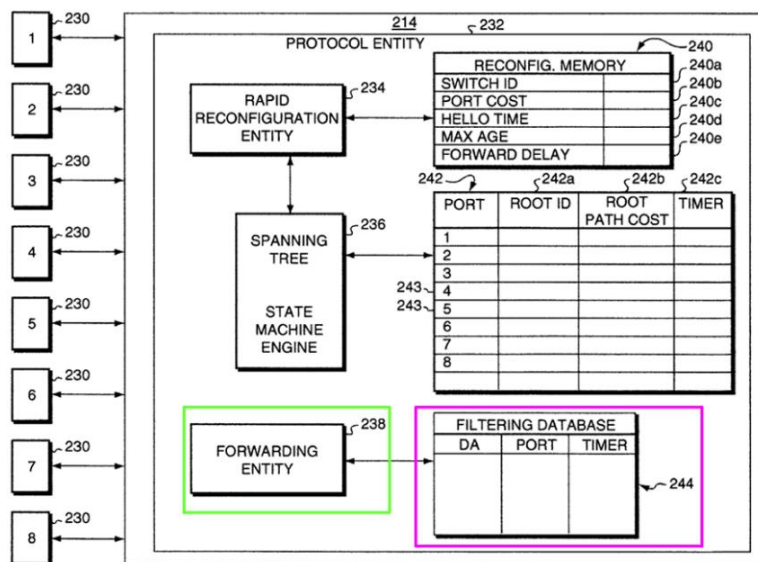


FIG. 2
Annotated Figure 2 of Gai

158. Gai discloses building filtering database 244 using information from traffic received at a port in an active state (*i.e.*, a forwarding state). In operation, ports 230 receive messages (*i.e.*, traffic) and pass them along to forwarding entity 238. EX1005, 11:16-18. Forwarding entity 238 looks up the destination address (“DA”) of the messages in the filtering database 244. If the destination address is found in the filtering database 244, a port (*e.g.*, port 4) corresponding to the destination address is identified. The message is switched out on the port (*e.g.*, port 4). EX1005, 11:19-23. If the destination address is not found in filtering database 244, the forwarding entity 238 looks up the source address in filtering

database 244. If the source address is not found in filtering database 244, a new entry is formed in filtering database 244 (*i.e.*, building filtering database 244).

EX1005, 11:24-33.

159. Gai discloses that each of its switches, including its backbone switches (hence the claimed “edge switch”), include filtering databases. EX1005, 1:53-54 (“This information is then stored by each switch in a block of memory referred to as a filtering database.”), 13:67-14:2 (“upon receipt of these multicast messages 400, the other devices within network 100 update their corresponding filtering databases”). Thus, limitation [1C] is satisfied.

e. [1D] responsive to a failure associated with the first link, placing the second port in the active state and the first port in the blocking state; and

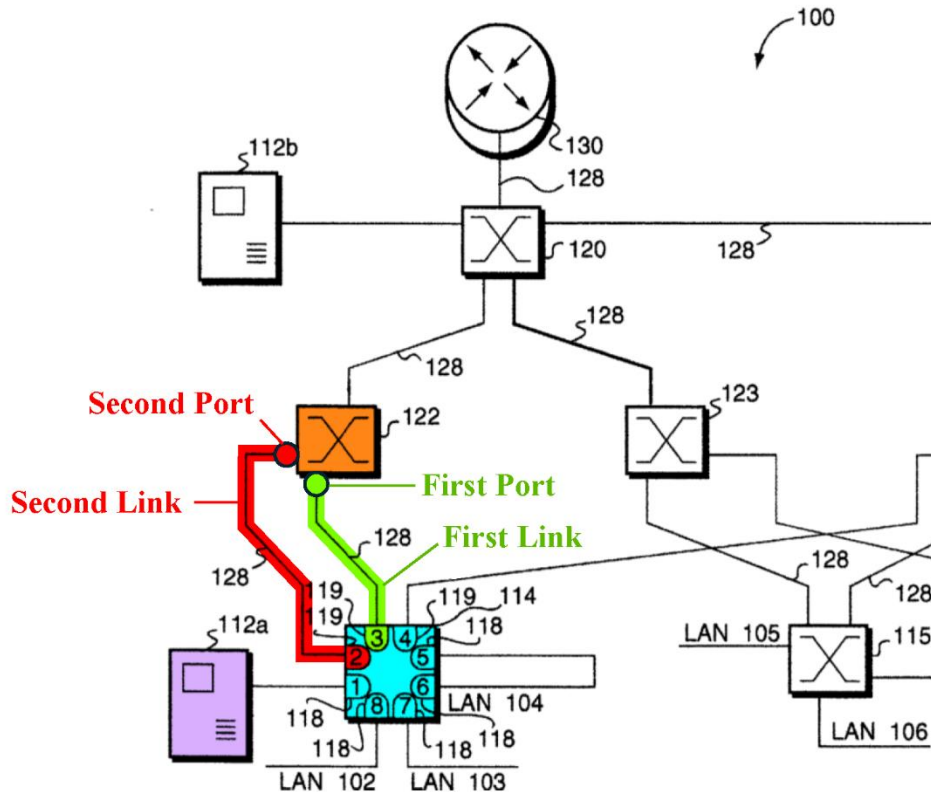
160. The '986 patent admits that Gai discloses detecting a failure associated with a link and thereafter placing a back-up port in the forwarding state to become the new active port. EX1001, 3:26-29 (“Upon detection of a failure at the active forwarding port, the switch immediately places one of the back-up trunk ports in the forwarding state, so that it becomes the new active port.”).

161. The '986 patent's admission is consistent with Gai's own disclosure. Gai discloses that “access switches 114-116 and backbone switches 120-125 detect failures.” EX1005, 11:64-65. Switches in Gai can detect failures using two different techniques. First, a switch (*e.g.*, an access switch or a backbone switch)

identifies a link failure “[i]f a switch stops receiving BPDUs on its root port.”

EX1005, 4:1-3. Second, a switch identifies a link failure by performing a link integrity test that “typically exchanges test messages across the respective link 128 at a relatively high rate (e.g., every ten milliseconds).” EX1005, 12:4-9. POSAs would have understood that a failed link between a backbone switch (*e.g.*, backbone switch 122) and an access switch (*e.g.*, access switch 114) would be recognized both by the backbone switch (as a downstream link failure) and by the access switch (as an upstream link failure). For example, IEEE 1999 explains this as noted above. EX1009, 166 (“the fault would be independently detected by switches both upstream as well as downstream of the fault”).

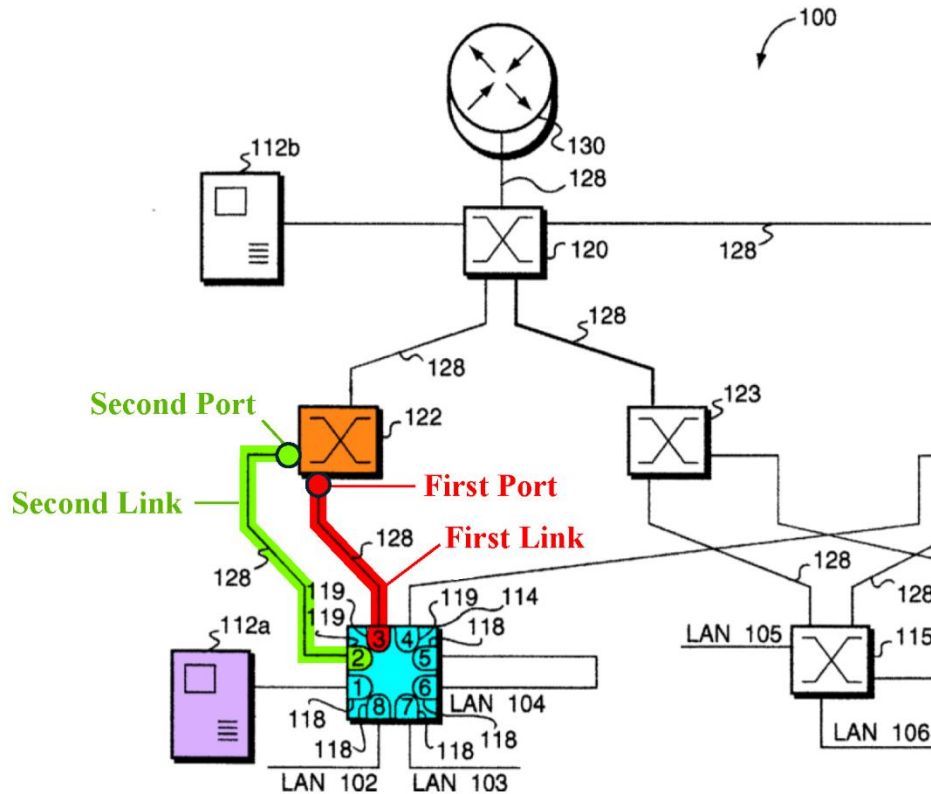
162. Gai discloses port reconfiguration responsive to a link failure. In an annotated version of Gai’s Figure 1 below showing the configuration before link failure, port 3 of access switch 114 is in the forwarding state and is connected to a corresponding first (active) port (identified by a green dot) of backbone switch 122 via an active link, which is the “active link” of [1D]. Port 2 of access switch 114 is in the blocking state and is connected to a corresponding second (blocked) port (identified by a red dot) of backbone switch 122 via the inactive link. The existence of these ports on backbone switch 122 is implicit, and certainly obvious, as previously explained.



Annotated Figure 1 of Gai before link failure

163. After detecting a failure of the active link, access switch 114 places its port 2 in the forwarding state and its port 3 in the blocking state. EX1005, 11:48-52; 14:40-51. The inactive (red) link shown above thus becomes the new active link, which is green below in another annotated version of Gai's Figure 1. The first port on backbone switch 122 (connected to port 3 on access switch 114) is placed in the blocking state (identified below by a red dot), and the active link shown above becomes an inactive link in red below. The second port on the backbone switch 122 (connected to port 2 on access switch 114) is placed in the forwarding state (*i.e.*, the active state) (identified below by a green dot). The

existence of these ports on backbone switch 122 is implicit, and certainly obvious, as previously explained.



Annotated Figure 1 of Gai after link failure

164. As explained above in [1A], backbone switch 122 qualifies as the claimed “edge switch.” Also, as explained above in [1A], backbone switch 122 operates like access switch 114 such that backbone switch 122 can transition a port between the forwarding and blocking states. As explained immediately above in this subsection, backbone switch 122 can detect a failure associated with the active link and, responsive to such a failure, backbone switch 122 places a port previously in the blocking state (*i.e.*, its second port as identified above with respect to [1B])

in the forwarding state and places another port previously in the active state (*i.e.*, its first port as identified above with respect to [1B]) in the blocking state, meeting and/or rendering obvious [1D].V.A.4.cV.A.4.e

- f. **[1E] sending dummy traffic from the edge switch over the second link to the one or more access switches, so as to cause each of the one or more access switches to modify its respective database responsive to the second port being in the active state.**

165. The '986 patent and Gai send the same dummy traffic between switches. The '986 patent admits that Gai discloses sending dummy multicast messages (*i.e.*, dummy traffic) upstream using a root port after detection of a link failure and, “[u]pon receiving these messages, other devices in the network learn to use the new active port.” EX1001, 3:26-35.

166. Gai’s dummy multicast messages meet [1E]’s “dummy traffic.” Gai’s Figure 4 (reproduced below) shows a dummy multicast message 400 that “includes a destination address (DA) field 410 and a source address (SA) field 412 that may be compatible with the Media Access Control (MAC) layer of the protocol stack.” EX1005, 13:1-4. Gai discloses generating a dummy multicast message by “load[ing] SA field 412 with the address of an entity directly coupled to switch 114.” EX1005, 13:14-16. The address of an entity directly coupled to switch 114 would be in the filtering database of switch 114. EX1005, 8:31-39 (“filtering database 244 configured to store address information corresponding to the entities

of network 100”). Loading SA field 412 in this way matches the ’986 patent, which describes “[f]or every entry in the filtering database of the edge switch that is not associated with the failed port...the edge switch sends a dummy frame...with a source MAC address equal to the destination MAC address listed for that entry.” EX1001, 9:44-49. Put another way, the source MAC address of the dummy frame in the ’986 patent is one of the destination MAC addresses in a filtering database, which is the same as Gai loading an address from a filtering database into SA field 412. Thus, a source address of Gai’s dummy multicast message is the same as the source address of the dummy frame of the ’986 patent. Gai also discloses loading the DA field “with a multicast address that causes message 400 to be received by all devices (e.g., access switches 114-116 and backbone switches 120-125) within the network 100.” EX1005, 13:9-13. This matches the ’986 patent, which explains that “[t]he destination MAC address of the dummy frame is a multicast address or other special address selected to ensure that the dummy frame is flooded to all the switches downstream from the edge switch.” EX1001, 9:53-56.

167. In summary, the dummy multicast message disclosed in Gai meets the dummy traffic of [1E] because they are both data messages that are loaded with source and destination addresses in the same manner.

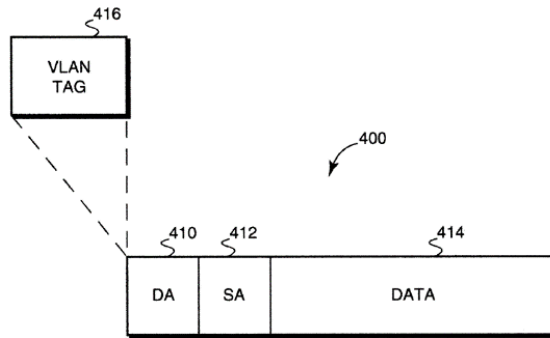


FIG. 4

Figure 4 of Gai

168. Gai discloses that dummy multicast messages can be sent by any switch. Figure 2 of Gai (annotated version reproduced below) shows a switch 214 that is described as corresponding to access switch 114. EX1005, 7:62-64.

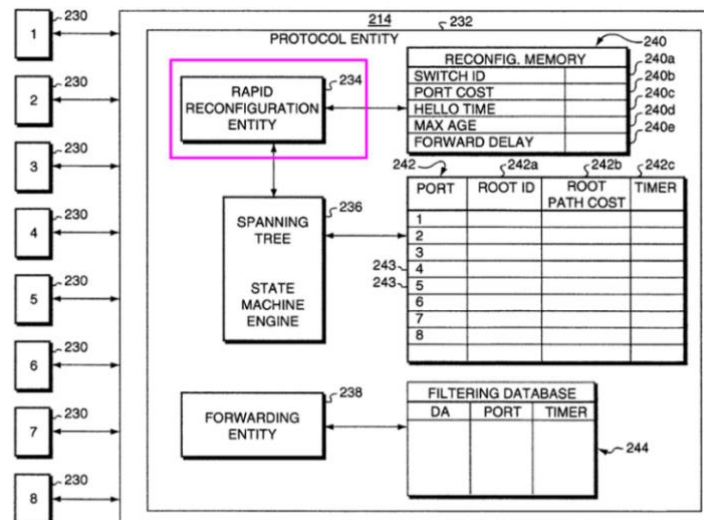


FIG. 2

Annotated version of Figure 2 of Gai

169. The switch implementation shown in Gai's Figure 2 includes a rapid reconfiguration entity 234. EX1005, 8:12-14. Rapid reconfiguration entity 234 is Gai's dummy multicast message mechanism that informs other devices of a new

forwarding port by transmitting dummy multicast messages through the new port. EX1005, 12:59-66. Rapid reconfiguration entity 234 is also implemented by at least backbone switch 120. EX1005, 8:66-9:3 (“the rapid reconfiguration entity 234 of the selected backbone switch 120”). Therefore, the switch implementation shown in Figure 2, which generates and transmits dummy multicast messages using rapid reconfiguration entity 234, is also applicable to backbone switch 120. Thus, POSAs would have understood that the switch implementation shown in Figure 2 of Gai is applicable to any switch in Gai, or at least that it would have been obvious to use it with any switch in Gai since the rapid reconfiguration entity 234 is described as being implemented at other switches in the network, and there is no technical barrier to using it in every switch. Therefore, POSAs would have understood that both access switches and backbone switches in Gai can generate and transmit dummy multicast messages.

170. The purpose and benefit of Gai’s dummy multicast message and the dummy traffic in [1E] are the same. Gai discloses that “[u]pon transition to the new forwarding port, the device begins transmitting ‘dummy’ multicast messages through the new port” (*i.e.*, “responsive to the second port being in the active state,” as claimed). EX1005, 5:54-56.

171. Gai further discloses that through dummy multicast messages, “other devices within the network learn to utilize the new forwarding port, rather than the

failed port, when directing messages to these entities”—reconfiguring the network “substantially sooner than the time required by the conventional spanning tree algorithm while still avoiding the formation of loops.” EX1005, 5:65-6:5; 6:50-54. In Gai, when other devices within the network receive dummy multicast messages, they enter this new information in their respective filtering databases. EX1005, 13:34-38. This matches the ’986 patent, which says “[u]pon receiving these dummy frames, the access switches use the standard MAC learning algorithm to rebuild their own filtering databases so that upstream traffic is immediately directed to the new active link,” purportedly completing network reconfiguration in less than one second. EX1001, 4:24-30.

172. Gai discloses sending dummy multicast messages to upstream devices after a link failure. As an example, Gai discloses access switch 114 selecting port 4 as a new root port after a link failure, as shown in annotated Figure 1 of Gai below. Access switch 114 then sends a dummy multicast message with a source address of server 112a through a new active link to backbone switch 124. EX1005, 13:14-23. Backbone switch 124 updates its filtering database by replacing previous information for server 112a with information referencing new port 4 of access switch 114. EX1005, 13:34-38.

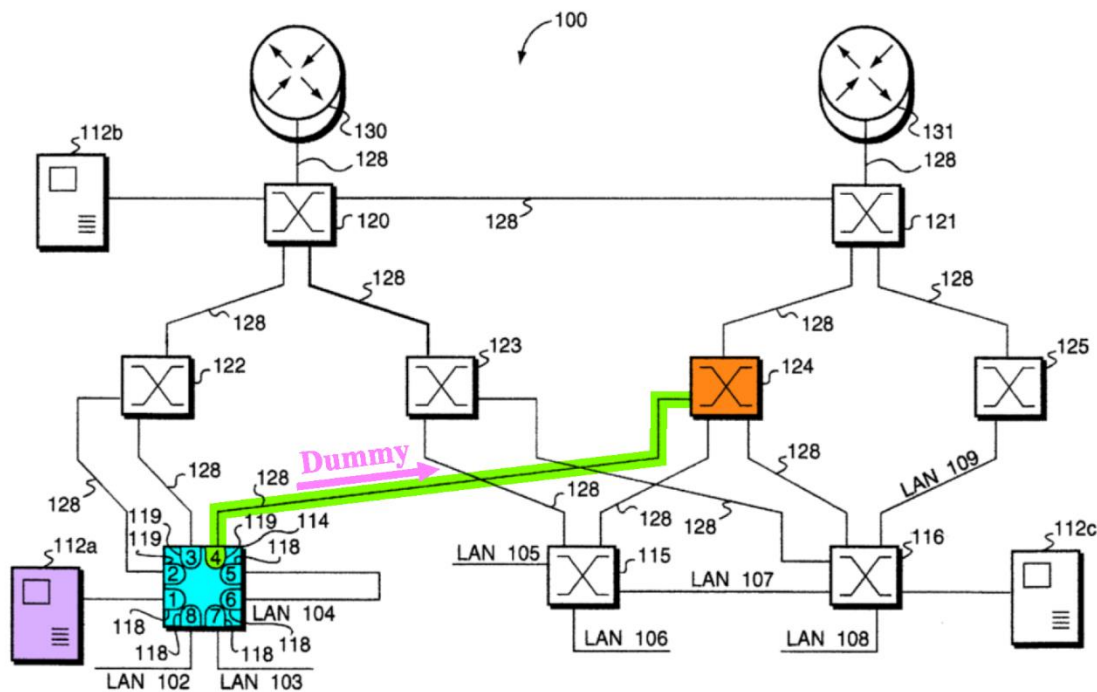


FIG. 1

Figure 1 of Gai: upstream dummy traffic

173. With reference to downstream directionality, it would have been obvious to POSAs to send dummy multicast messages both upstream, as explained above, as well as downstream, for the benefit of reducing the total link failure recovery time because the link failure can be addressed by upstream and downstream switches in parallel. As explained above, POSAs would have been motivated, by IEEE 1999, to modify Gai to send dummy traffic downstream as well as upstream to achieve this benefit, and would have enjoyed a reasonable expectation of success in doing so.

174. In summary, when modified in view of IEEE 1999's disclosure that it was advantageous to send self-healing messages both upstream and downstream, Gai's backbone switches send Gai's dummy traffic both upstream and downstream, regardless of whether a link failure is detected upstream or downstream of the switch. Sending dummy traffic both upstream and downstream has the known benefit of reducing the total link recovery time since the failure notification is propagated in all directions substantially simultaneously. Thus, modified Gai's backbone switch (the claimed "edge switch") sends dummy traffic downstream to Gai's access switch, causing the access switch to update its filtering database in the same manner as if backbone switch 122 received a dummy message from access switch 114. These effects occur in response to the second port being placed in the active state, after a link failure is detected, thus satisfying every portion of [1E].

175. In short, Gai renders obvious claim 1 in view of IEEE 1999.

5. **Claim 2: A method according to claim 1, wherein placing the second port in the blocking state comprises configuring the first and second ports so that no loop is created in the network between the edge switch and the one or more access switches.**

176. Gai discloses switches that execute a spanning tree algorithm, which calculates a spanning tree that is loop-free. EX1005, 2:30-41; 7:59-61. As explained *supra* §III.B.3., conventional STP assigns roles to ports to prevent

network loops, which indicates that claim 2 is conventional STP. For each switch, only one root port is placed in the forwarding state while the remaining root ports on that switch are blocked. EX1005, 6:33-38; 11:8-12. Accordingly, for an access switch that is connected to a backbone switch, only one access switch port connected to the backbone switch is in the forwarding state while the other access switch ports connected to the backbone switch are blocked. Gai explains that configuring only one port on each access switch for connectivity to the backbone switch avoids loops. EX1005, 12:43-52.

177. As explained above in [1A], Gai implicitly discloses that backbone switch 122 has ports that are coupled to ports 2 and 3 of access switch 114. Therefore, the configuration of the ports of backbone switch 122 meet claim 2. As explained above in [1B]V.A.4.c, Gai discloses, or at least renders obvious, placing the “first” port of backbone switch 122 that is connected to port 3 of access switch 114 in the active state while placing the “second” port of backbone switch 122 that is connected to port 2 of access switch 114 in the blocking state. Among the designated ports of backbone switch 122 that are connected to access switch 114, only the designated port that is connected to root port 3 of access switch 114 is in the forwarding state, while the remaining designated port(s) connected to access switch 114 are blocked. EX1005, 6:33-35. By placing only one designated port of backbone switch 122 that is connected to access switch 114 in the forwarding state

while placing other designated ports that are connected to access switch 114 in the blocking state, no loop is created in the network between backbone switch 122 and access switch 114.

6. Claim 3: A method according to claim 2, wherein configuring the first and second ports comprises configuring the edge switch so as to decrease a radius of a spanning tree established in the network.

178. The '986 patent describes reducing a radius of a spanning tree by running the standard spanning tree algorithm over fewer branching communication links. EX1001, 4:44-48. As explained above in §III.A.2., reducing a radius of the spanning tree refers to reducing the number of switches to be informed of a topology change. The '986 patent reduces the number of such links by managing redundant downstream ports in a purportedly novel manner. EX1001, 4:35-37. The '986 patent manages redundant downstream ports by placing one in the forwarding state while blocking the others. EX1001, 3:67-4:4. Therefore, the '986 patent decreases a radius of a spanning tree by reducing the number of downstream switches that need to be informed of a topology change; a downstream access switch connected to a port in a forwarding state is informed while another downstream access switch connected to a port in a blocking state is not informed. Likewise, Gai discloses, or at least renders obvious, placing one port of backbone switch 122 that is connected to the root port of access switch 114 in the forwarding state while blocking other backbone switch ports connected to the access switch

114 (*supra* [1B]V.A.4.c), which—for the same reasons explained in the ’986 patent—decreases a radius of a spanning tree in Gai’s network, thus meeting claim 3.

7. **6. Claim 6: A method according to claim 1, wherein sending the dummy traffic comprises sending data frames originating at the edge switch while having source addresses corresponding to network addresses of other entities in the network that are accessible to the access switches via the edge switch.**

179. As explained above in [1A], backbone switch 122 is an edge switch. As explained above in [1E]V.A.4.f, it would have been obvious for backbone switch 122 (*i.e.*, the edge switch) to send dummy multicast messages and thereby originate them. Such “dummy multicast messages carry the source address of each entity that is directly coupled to the device with the new active port or downstream thereof (relative to the root) and are received by other devices in the computer network.” EX1005, 5:54-60. Since such entities directly coupled to the device (*e.g.*, backbone switch 122) or downstream thereof are entities in the network and are accessible to access switches via the device, Gai’s backbone switch 122 sends the dummy traffic as claimed.

8. **Claim 7: A method according to claim 6, wherein sending the data frames comprises copying the source addresses for the data frames from the database of the edge switch.**

180. Gai discloses that each switch (*e.g.*, backbone switch 122) has a filtering database. EX1005, 1:53-54. Gai discloses that “filtering database 244 [is]

configured to store [addresses] corresponding to the entities of network 100.”

EX1005, 8:31-38. Gai discloses a switch loading a source address field of a dummy multicast message 400 with “the address of an entity directly coupled to switch 114.” EX1005, 13:1-4; 13:14-16; 13:50-57. Since a switch stores these addresses in its filtering database 244, it would have been obvious—as the simplest and fastest way of obtaining the address—for the switch to load (*i.e.*, copy) an address from its filtering database 244 into the source address field of the dummy multicast message 400, meeting claim 7. Because the switch stores these addresses in its filtering database 244, a POSA would find it redundant if these addresses were duplicated elsewhere in the switch. Therefore, it would be obvious to POSAs that the source addresses for the data frames come from the disclosed location of the addresses, *i.e.*, the filtering database 244 to meet claim 7.

9. Claim 8: A method according to claim 7, wherein

a. [8A] the edge switch has three or more ports, including the first and second ports, and

181. As explained above in [1A], backbone switch 122 is an “edge switch”. As explained above, backbone switch 122 has a plurality of ports and, in the case of FIG. 1 of Gai, at least three ports. Therefore, Gai meets, or at least renders obvious, [8A] because backbone switch 122 is an edge switch and, as shown below in an annotated version of Figure 1 of Gai, has at least three ports (identified by the yellow dots): a first port connected to port 3 of access switch 114 (the “first” port),

a second port connected to port 2 of access switch 114 (the “second” port), and a third port connected to backbone switch 120.

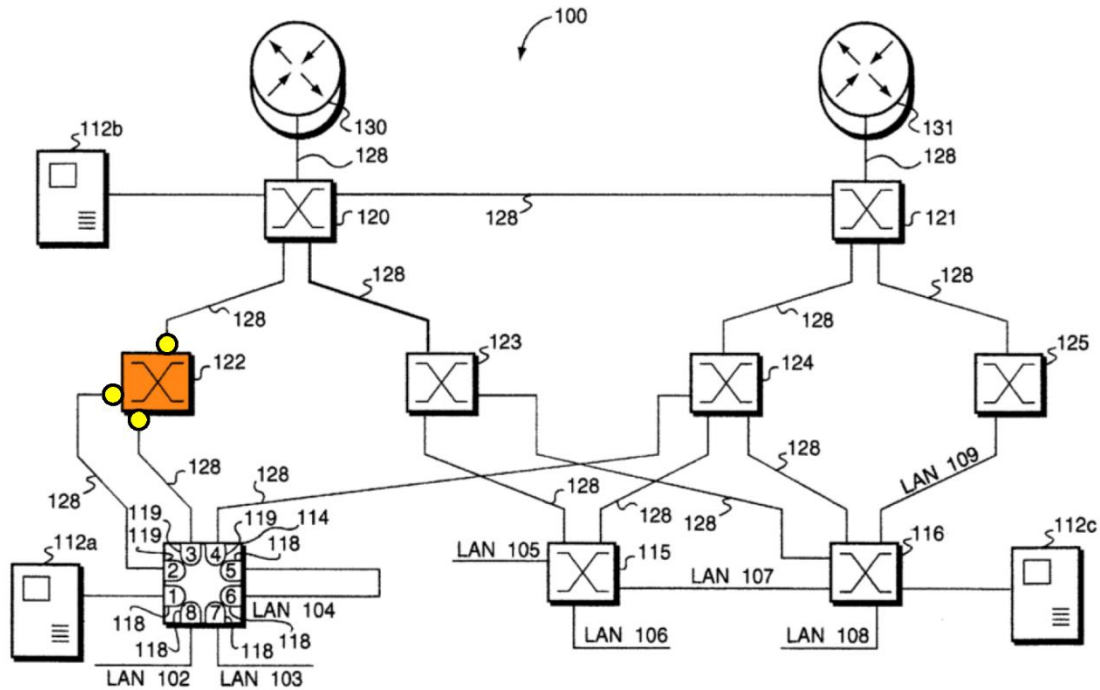


FIG. 1

Annotated version of Figure 1 of Gai

- b. [8B] wherein prior to the failure, the database comprises entries associated with a plurality of the ports, and**

182. Gai meets [8B] because backbone switch 122, like every switch, has a filtering database 244 that comprises entries associated with a plurality of ports of backbone switch 122. EX1005, 1:53-54. As shown below in an annotated version of Figure 2 of Gai, filtering database 244 has a “plurality of columns or cells, including a destination address cell, a destination port cell and a corresponding

timer cell.” EX1005, 8:33-37; 1:53-54. “Each row or entry in the filtering database 244 preferably corresponds to a particular entity.” EX1005, 8:37-39.

183. The destination address column (identified by “DA” enclosed in the pink box) stores destination addresses. Each row or entry in the destination port column (enclosed in the green box) stores destination port information for a plurality of ports (shown as ports 230 in Figure 2 below and enclosed in the red box) for backbone switch 122, and each row or entry is thereby associated with the plurality of the ports, meeting [8B].

184. In operation, the switch 214 may receive a message on one of its 8 ports (identified by 1-8). The message includes a destination address. The forwarding entity 238 looks up the destination address in the destination address column of the filtering database 244. If the destination address is found in the filtering database 244, then the forwarding entity 238 looks up what port is listed in the destination port column for the destination address. The port listed in the destination port column is one of ports 1-8 shown in Figure 2 below. The forwarding entity 238 then switches out the message onto the identified port. Therefore, the filtering database 244 shown in Figure 2 below comprises entries in the destination port column associated with a plurality of ports (*i.e.*, ports 1-8 shown in Figure 2) to meet [8B].

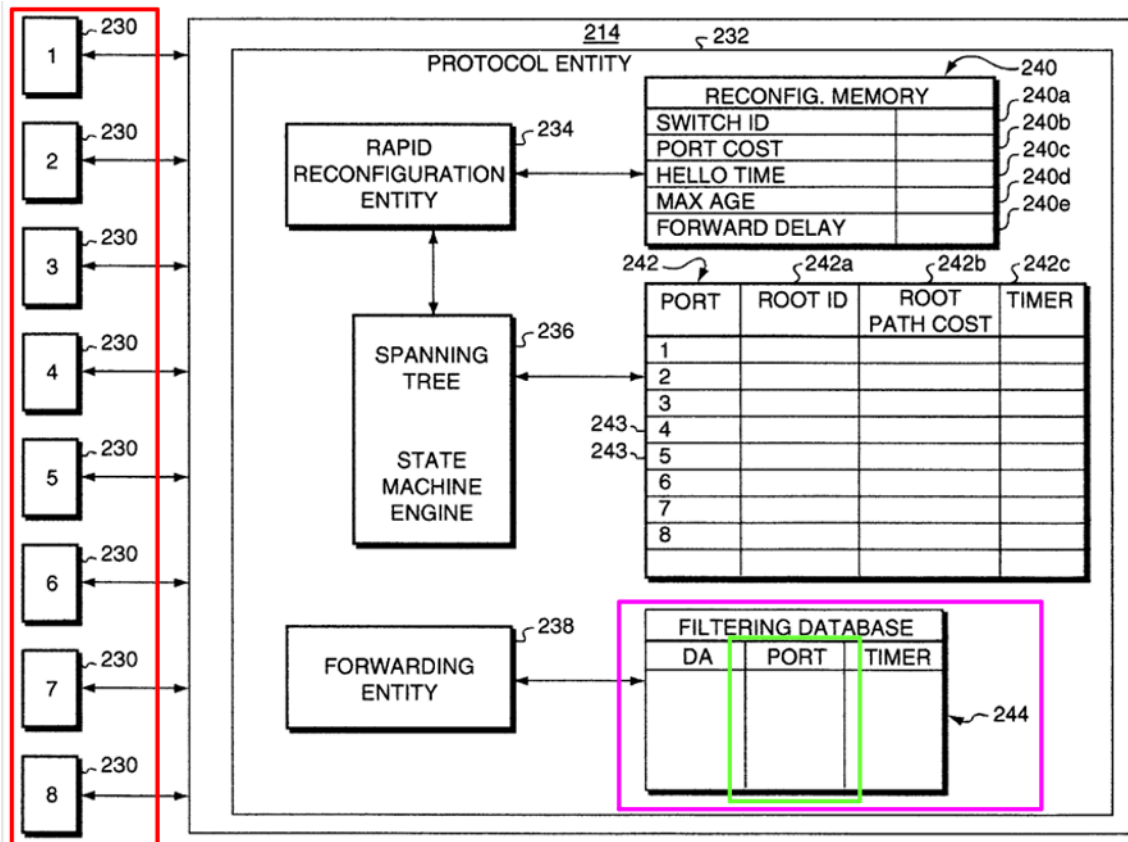


FIG. 2

Annotated version of Figure 2 of Gai

- c. **[8C] wherein copying the source addresses comprises using the network addresses in all the entries that are not associated with the first port as the source addresses of the data frames.**

185. As discussed in [8B]V.A.9.b, Gai discloses using the network addresses in filtering database 244 as the source addresses of dummy multicast messages. Gai further discloses “generat[ing] and transmit[ing] dummy multicast messages 400 for all entities directly coupled to switch 114 or downstream thereof (relative to the root) through the new forwarding port (i.e., port number four).” EX1005, 14:55-58. Each dummy multicast message has a source address

corresponding to one such entity. EX1005, 13:1-4; 13:14-33. As explained above in [1E] and claim 7, sending the dummy messages from an edge switch would have been obvious. V.A.4.f A device that receives the dummy multicast message, which corresponds to one such entity, notes that the message is received from a different source port than the receiving device previously associated with the entity. It would be obvious to POSAs not to use source addresses associated with the failed port as that would provide incorrect topology change information to the network. Instead, to provide the correct topology change information to the network (i, it would be obvious to POSAs to use source addresses not associated with the failed port. As such, each dummy multicast message has a network address associated with the new forwarding port and not the previous forwarding port, which meets [8C].

10. Claim 9: A method according to claim 6, wherein sending the data frames comprises incorporating a Virtual Local Area Network (VLAN) tag in a header of the data frames.

186. Gai discloses incorporating a virtual local area network (VLAN) tag 416 (red box below) to a header of a dummy multicast message 400 as shown below in an annotated version of Gai's Figure 4, meeting claim 9. EX1005, 17:32-41.

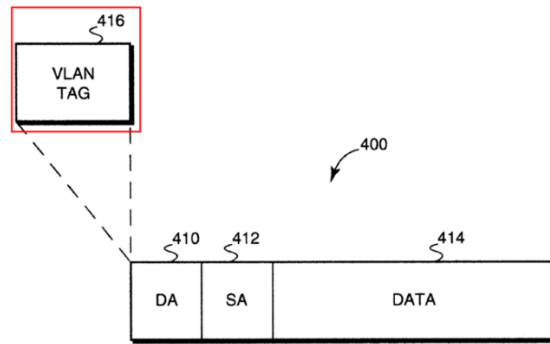


FIG. 4

Annotated version of Figure 4 of Gai

- 11. Claim 10: A method according to claim 6, wherein sending the data frames comprises multicasting the data frames to all of the one or more access switches.**

187. Gai discloses “load[ing] DA field 410 with a multicast address that causes [dummy] message 400 to be received by all devices (*e.g.*, access switches 114-116...) within the network 100,” thus meeting claim 10. EX1005, 13:9-13.

- 12. Claim 11: A method according to claim 1, wherein sending the dummy traffic comprises sending data frames downstream from the edge switch toward the access switches.**

188. As explained in [1E]V.A.4.f, it would have been obvious for all of Gai’s switches, including backbone switch 122 (the claimed “edge switch”) to send dummy messages, including downstream from backbone switch 122 toward access switch 114. Per the ’986 patent, the claimed “dummy traffic” refers to dummy data frames. EX1001, 5:24-25, 12:51-52. Data frames are messages. EX1005, 11:16. Therefore, Gai’s “dummy messages” are “dummy traffic” comprising “data frames,” meeting claim 11.

13. Claim 12

- a. **[12A] A method for network communication, comprising: coupling a first bridge in a network to a second bridge located downstream of the first bridge, by connecting at least first and second redundant links to respective first and second ports of the first bridge so as to communicate with the second bridge;**

189. Gai meets [12A] for the same reasons as [1A]. The first and second bridges of [12A] are met by backbone switch 122 and access switch 114, respectively. Backbone switch 122 and access switch 114 are coupled together in the claimed arrangement as shown by Figure 1 of Gai. As shown by Figure 1 of Gai, backbone switch 122 is upstream of access switch 114. As discussed in [1A], backbone switch 122, like access switch 114, has multiple ports (*i.e.*, at least two ports), or at least that would be obvious. As discussed in [1A] and shown in Figure 1 of Gai, backbone switch 122 is coupled (and at least is obviously coupled) to access switch 114 by connecting first and second ports of backbone switch 122 to first and second ports of access switch 114 using redundant links to meet [12A].

- b. **[12B] placing the first port in an active state, while placing the second port in a blocking state;**

190. Gai meets [12B] for the same reasons as [1B]. As explained above in [1B], Gai implicitly discloses (and renders obvious) placing a first port of backbone switch 122 in a forwarding state (*i.e.*, an active state) and placing a

second port of backbone switch 122 in a blocking state like access switch 114 places port 3 in the forwarding state and port 2 in the blocking state to meet [12B].

- c. **[12C] conveying communication traffic over the first link while the first port is in the active state, thereby causing the bridges to build respective databases for use in forwarding the traffic;**

191. Gai meets [12C] for the same reasons as [1C]. As explained above in [1C], Gai discloses (and renders obvious) that every switch, such as backbone switch 122 and access switch 114, has a filtering database for use in forwarding traffic to meet [12C].

- d. **[12D] responsive to a failure associated with the first link, placing the second port in the active state and the first port in the blocking state; and**

192. Gai meets [12D] for the same reasons as [1D]. As discussed in [1D], access switch 114, responsive to a failure, places port 2 in the forwarding state (*i.e.*, the active state) and port 3 in the blocking state. As discussed in [1D], backbone switch 122 operates like access switch 114 such that the port of backbone switch 122 coupled to port 2 of access switch 114 is placed in the forwarding state and the port of backbone switch 122 coupled to port 3 of access switch 114 is placed (or it would be obvious to place it) in the blocking state to meet [12D].

- e. **[12E] sending dummy traffic from the first bridge over the second link downstream to the second bridge,**

so as to cause the second bridge to modify its database responsive to the second port being in the active state.

193. Gai meets [12E] for the same reasons as [1E]. As explained in [1E], Gai discloses that dummy traffic can be sent from any switch, including backbone switch 122, because every switch includes rapid reconfiguration entity 234 of FIG. 2 of Gai, which generates and transmits dummy multicast messages. As explained above in claim 11, “dummy messages” are “dummy traffic.” Therefore, Gai discloses sending dummy traffic from backbone switch 122 (*i.e.*, the first bridge) downstream to the access switch 114 (*i.e.*, the second bridge). Further, as discussed above in [1E], the dummy traffic is sent so as to cause a receiving bridge, such as access switch 114, to modify its filtering database in accordance with the root port change of backbone switch 122 to meet [12E].

14. Claim 13

- a. [13A] A network switching device, comprising: a network port, adapted to be coupled over a network trunk link to a core switch in a network so as to exchange communication traffic with the core switch;**

194. Gai discloses [13A] because backbone switch 122 meets the claimed network switching device. Backbone switch 122 comprises (or at least renders obvious, *supra* [1B]) a network port (*i.e.*, a trunk port) that is coupled to backbone switch 120 (*i.e.*, a core switch) in the network via a network trunk link to exchange communication traffic with the backbone switch 120. The '986 patent uses “trunk

port” synonymously with “root port.” EX1001, 3:23-24 (“[T]he trunk ports are on the upstream side of each switch, i.e., the side leading toward the root.”). This port on backbone switch 122 is coupled to backbone switch 120, which is a “core switch” per the ’986 patent because it connects access switches to a WAN via router(s). EX1001, 1:32-35; EX1005, Fig. 1. The purpose of the connection is to exchange communication traffic. EX1005, 7:31-34. Accordingly, Gai’s backbone switch 122 meets or renders obvious [13A].

- b. [13B] first and second access ports, adapted to be coupled via first and second redundant network access links to one or more access switches serving network users; and**

195. Gai meets [13B] for the same reasons as [1A]. As discussed in [1A], backbone switch 122 has (or at least it would have been obvious for it to have) first and second ports coupled to access switch 114 via first and second redundant network access links 128.

- c. [13C] a protocol processor, adapted to place the first access port in an active state, while placing the second access port in a blocking state, so that the communication traffic is conveyed over the first link while the first port is in the active state, thereby causing each of the access switches to build a respective access switch database for use in forwarding the traffic, the protocol processor being further adapted, responsive to a failure associated with the first link, to place the second access port in the active state and the first port in the blocking state and to send dummy traffic through the second access port over the second link to the one or more access**

switches, so as to cause each of the one or more access switches to modify its respective access switch database responsive to the second access port being in the active state.

196. The capabilities of the “protocol processor” of [13C] match the steps recited [1B]-[1E]. Gai thus renders obvious [13C] for the same reasons set forth in [1B]-[1E], and because Gai discloses protocol entity 232 (a “protocol processor”) for performing these functions, including via rapid reconfiguration entity 234, logic circuits and “programmable processing elements.” EX1005, 8:12, 8:40-51. These elements parallel the ’986 patent’s protocol processor 44, about which the ’986 patent explains: “[A]lthough protocol processor 44 is shown here as a single, separate entity, its functions are in actuality distributed among a number of separate processors...” EX1001, 8:58-61.

197. Switch 214, containing protocol entity 232, corresponds to access switch 114, but just as any switch in Gai can include a rapid reconfiguration entity 234 (EX1005, 8:12-15) it would have been obvious to include protocol entity 232 in other switches, such as backbone switches, since rapid reconfiguration entity 234 is part of protocol entity 232. EX1005, 8:67-9:1. In summary, [13C] is obvious for the same reasons as [1B]-[1E].

15. Claim 14: A device according to claim 13, wherein the protocol processor is adapted to configure the first and second access ports so that no loop is created in the network

between the edge switch and the one or more access switches.

198. There appears to be a typographical error in claim 14. To the extent “edge switch” in claim 14 is intended to refer to the “network switching device” of claim 13, Gai renders obvious claim 14, via Gai’s protocol entity 232, for the same reasons explained above with respect to claim 2. As discussed in claim 2, conventional STP assigns roles to ports to prevent network loops, which indicates that claim 14 is conventional STP.

16. Claim 15: A device according to claim 14, wherein the protocol processor is further adapted to configure the first and second access ports so as to decrease a radius of a spanning tree established in the network.

199. To the extent “edge switch” in claim 14 refers to claim 13’s “network switching device,” Gai renders obvious claim 15, via Gai’s protocol entity 232, for the same reasons explained above with respect to claim 3.

17. Claim 18: A device according to claim 13, wherein the dummy traffic comprises data frames originating at the edge switch while having source addresses corresponding to network addresses of other entities in the network that are accessible to the access switches via the edge switch.

200. To the extent “edge switch” in claim 18 refers to the “network switching device” of claim 13, Gai renders obvious claim 18, via Gai’s protocol entity 232, for the same reasons explained above with respect to claim 6. As explained above in [1A], backbone switch 122 is an edge switch. As explained

above in [1E]V.A.4.f, it would have been obvious for backbone switch 122 (*i.e.*, the edge switch) to send dummy multicast messages and thereby originate them. Such “dummy multicast messages carry the source address of each entity that is directly coupled to the device with the new active port or downstream thereof (relative to the root) and are received by other devices in the computer network.” EX1005, 5:54-60. Since such entities directly coupled to the device (*e.g.*, backbone switch 122) or downstream thereof are entities in the network and are accessible to access switches via the device, Gai’s backbone switch 122 meets the device of claim 18.

18. Claim 19: A device according to claim 18, and

a. [19A] comprising a memory,

201. Gai’s protocol entity 232 meets [19A] because it comprises “memory 240 and tables 242 and 244 [that] may be implemented as content addressable memory (CAM) devices.” EX1005, 8:40-42, 1:53-54.

b. [19B] wherein the protocol processor is adapted to build an edge switch database in the memory responsive to the communication traffic prior to the failure associated with the first link, and to copy the source addresses for the data frames from the edge switch database.

202. To the extent “edge switch” in [19B] refers to the “network switching device” of claim 13, Gai’s protocol entity 232 meets [19B] for the same reasons explained above with respect to [1A], [1C], [13C], and claim 7. As discussed in

[1A], backbone switch 122 is an “edge switch.” As discussed in [13C], protocol entity 232, which includes memory that meets [19A], of Gai is in all switches including backbone switch 122. Further, as discussed in [13C], protocol entity 232 meets the claimed protocol processor. As discussed in [1C], all switches including backbone switch 122 have filtering databases. As discussed in claim 7, source addresses for the dummy multicast messages are copied from the filtering database of the backbone switch 122. Therefore, Gai discloses that the protocol entity 232 of backbone switch 122 builds a filtering database from which source addresses are copied for dummy multicast messages to meet [19B].

19. Claim 20: A device according to claim 19, wherein prior to the failure, the edge switch database comprises entries associated with a plurality of the ports, and wherein the protocol processor is adapted to use the network addresses in all the entries that are not associated with the first access port as the source addresses of the data frames.

203. To the extent “edge switch” in claim 20 refers to the “core switch” of claim 13, Gai renders obvious claim 20 for the reasons explained above with respect to claims 8 and 13. As discussed in [8B], switch 214, which includes 8 ports, is applicable to backbone switch 122 and thereby backbone switch 122 has a plurality of ports. As discussed in claim 13, protocol entity 232, which is included in switch 214 and thereby included in backbone switch 122, meets the claimed protocol processor. As discussed in [8C], protocol entity 232 uses source addresses associated with the new root port and not associated with the failed root

port to generate dummy multicast messages 400. Therefore, Gai discloses backbone switch 122, which includes protocol entity 232 and a plurality of ports, as generating dummy multicast messages 400 using source addresses not associated with the failed root port to meet claim 20.

20. Claim 21: A device according to claim 18, wherein the protocol processor is adapted to incorporate a Virtual Local Area Network (VLAN) tag in a header of the data frames.

204. Gai renders obvious claim 21 for the same reasons explained above with respect to claim 9. As discussed in claim 9, Gai discloses incorporating a virtual local area network (VLAN) tag 416 to a header of a dummy multicast message 400 as shown in Gai's Figure 4, meeting claim 9. EX1005, 17:32-41.

21. Claim 22: A device according to claim 18, wherein the protocol processor is adapted to multicast the data frames to all of the one or more access switches.

205. Gai renders obvious claim 22, via its protocol processor, for the same reasons explained above with respect to claim 10. As discussed in claim 10, Gai discloses "load[ing] DA field 410 with a multicast address that causes [dummy] message 400 to be received by all devices (*e.g.*, access switches 114-116...) within the network 100," thus meeting claim 22. EX1005, 13:9-13.

22. Claim 23: A device according to claim 13, wherein the dummy traffic comprises data frames sent downstream from the edge switch toward the access switches.

206. Gai renders obvious claim 23 for the same reasons explained above with respect to claim 11. As explained in [1E], it would have been obvious for all of Gai's switches, including backbone switch 122 (the claimed "edge switch") to send dummy messages, including downstream from backbone switch 122 toward access switch 114.

23. Claim 24

- a. [24A] A bridge device for use in a communication network having a spanning tree root, the device comprising: a designated port, adapted to be coupled over an upstream link in the network so as to communicate with the root;**

207. Gai discloses backbone switch 122 (*i.e.*, the claimed bridge device) in a network topology that meets [24A]. The network topology includes backbone switch 120 as the spanning tree root. EX1005, 8:57-59. Further, as provided in [1A], backbone switch 122 has (or such would be obvious) a designated port coupled to access switch 114 over link 128 (an upstream link), allowing access switch 114 to communicate (via backbone switch 122) with the root. EX1005, Fig. 1.

- b. [24B] first and second downstream ports, adapted to be coupled via first and second redundant links to a**

further bridge device located downstream in the network relative to the spanning tree root; and

208. As explained in [1A], backbone switch 122 has (or such would be obvious) first and second downstream ports coupled to access switch 114 (*i.e.*, a further bridge device) over redundant links 128 to meet [24B]. As discussed in [1A], backbone switch 122, like access switch 114, has multiple ports (*i.e.*, at least two downstream ports). As discussed in [1A] and shown in Figure 1 of Gai, backbone switch 122 is coupled to access switch 114 by connecting first and second ports (*i.e.*, first and second downstream ports) of backbone switch 122 to first and second ports of access switch 114 using redundant links to meet [24B].

- c. **[24C] a protocol processor, which is adapted to place the first downstream port in an active state, while placing the second downstream port in a blocking state, so that communication traffic is conveyed over the first downstream link while the first port is in the active state, thereby causing the bridge devices to build respective databases for use in forwarding the traffic, the protocol processor being further adapted, responsive to a failure associated with the first link, to place the second downstream port in the active state and to send dummy traffic over the second link downstream to the further bridge device, so as to cause the further bridge device to modify its database responsive to the second downstream port being in the active state.**

209. Modified Gai renders obvious [24C], via Gai's protocol entity 232, for the same reasons explained above with respect to [13C]. As explained in [13C], the capabilities of the "protocol processor" of [13C] match the steps recited [1B]-

[1E]. Gai thus renders obvious [24C] for the same reasons set forth in [1B]-[1E], and because Gai discloses protocol entity 232 (a “protocol processor”) for performing these functions, including via rapid reconfiguration entity 234, logic circuits and “programmable processing elements.” EX1005, 8:12, 8:40-51. These elements parallel the ’986 patent’s protocol processor 44, about which the ’986 patent explains: “[A]lthough protocol processor 44 is shown here as a single, separate entity, its functions are in actuality distributed among a number of separate processors...” EX1001, 8:58-61.

210. Switch 214, containing protocol entity 232, corresponds to access switch 114, but just as any switch in Gai can include a rapid reconfiguration entity 234 (EX1005, 8:12-15) it would have been obvious to include protocol entity 232 in other switches, such as backbone switches, since rapid reconfiguration entity 234 is part of protocol entity 232. EX1005, 8:67-9:1. In summary, [24C] is obvious for the same reasons as [1B]-[1E] and [13C].

B. Ground 2: Claims 4, 5, 16, 17: Obvious over Gai in View of IEEE 1999, Further in View of Jain

1. U.S. Patent No. 6,330,229 (“Jain,” EX1006)

211. Jain discloses managing forwarding databases (*i.e.*, filtering databases) in case of a link failure. EX1006, Abstract, 6:52-54. Like Gai, Jain discloses detecting a failure and transitioning a backup port to take over for the failed port. EX1006, 5:15-18. To reduce recovery time, the forwarding databases

are updated by transferring information from the failed port to the backup port, which is now active. The updates are made before the change from the failed port to the backup port to maintain high network availability. EX1006, 4:28-33; 5:15-20.

2. Gai in View of IEEE 1999, Further in View of Jain

212. Gai and Jain are both directed to addressing link failures in spanning tree networks while reducing link failure recovery time. As discussed previously, Gai discloses an access switch with a protocol entity, which includes a filtering database in which each row or entry stores address information of a particular network entity. EX1005, 8:31-38, Fig. 2. As explained *supra* [13C], it would have been obvious to use the protocol entity in any backbone switch and, as explained *supra* [1A], Gai's backbone switches are "edge switches" as claimed. Therefore, it would have been obvious to use Gai's protocol entity in "edge switches." As discussed in Ground 1, sending dummy traffic downstream would have been obvious in view of IEEE 1999.

213. Gai discloses that, in response to a root port failure, the protocol entity immediately transitions a back-up port to be the new root port and starts processing messages through this newly-activated port. EX1005, 12:15-18, 12:32-42.

214. Gai expresses concern that messages may be misdirected or lost during the transition process. EX1005, 12:59-62, 15:18-21. POSAs would have

been motivated to modify Gai by incorporating Jain's technique of updating a filtering database responsive to a link failure for the benefit of reducing the possibility of misdirected/lost data messages during the self-healing operation. As modified, Gai's protocol entity would update the filtering database of a switch (including backbone switch 122) after detecting a failure at a port (*e.g.*, a downstream port). The update would be implemented by remapping entries from the failed port to a backup port (*e.g.*, a back-up downstream port). The update would be implemented before receiving communication traffic on the backup port to reduce the possibility of the backup port misdirecting or losing received messages using outdated database information. The modification would result in a backbone switch's protocol entity transitioning a failed port in the forwarding state to the blocking state and a backup port from the blocking state to the forwarding state, all with a reduced possibility of data messages being misdirected or lost because the backbone switch's database would be updated prior to notifying other network switches of the newly-activated backup port.

3. Claim 4: A method according to claim 1, and comprising, responsive to placing the second port in the active state, updating the database of the edge switch so that entries in the database point to the second port instead of the first port.

215. Gai transitions a backup port (*i.e.*, the second port) in the blocking state at a backbone switch (*e.g.*, backbone switch 122) to an active port in the

forwarding state (*i.e.*, the active state) after detecting a failure at a port (*i.e.*, the first port). *Supra* [1D]. Responsive to placing the backup port in the active state, Gai, modified in view of Jain, would update the filtering database of backbone switch 122 by transferring the database entries that pointed to the failed port to point instead to the backup port, thus rendering claim 4 obvious. EX1006, 5:15-20, 14:13-26.

4. **Claim 5: A method according to claim 4, wherein updating the database comprises changing the entries in the database to point to the second port before receiving the communication traffic from the one or more access switches on the second port.**

216. Modified in view of Jain, Gai would map, in backbone switch 122, the failed port forwarding data to the backup port after the backup port transitions from the blocking state to the forwarding state. EX1006, 14:13-15. “After mapping the forwarding data from the old root port to the new root port in block 703, a packet is sent on designated ports with the new root path cost data” to allow “descendant bridges to recompute their spanning tree configuration.” EX1006, 14:27-31. The descendant bridges (*e.g.*, one or more access switches) would not send communication traffic to the backup port until notified to do so because the descendant bridges would not have information in their respective filtering databases indicating that the backup port is active. The descendant bridges would only update their filtering databases to forward data to the backup port after

receiving the packet with the new root path cost data. Thus Gai, modified in view of Jain, meets claim 5 because it updates the filtering database in backbone switch 122 to point toward the backup port prior to the backup port receiving communication traffic from another network entity.

5. Claim 16: A device according to claim 13, and

a. [16A] comprising a memory,

217. Gai's protocol entity 232 meets [16A] because it comprises "memory 240 and tables 242 and 244 [that] may be implemented as content addressable memory (CAM) devices." EX1005, 8:40-42, 1:53-54.

b. [16B] wherein the protocol processor is adapted to build an edge switch database in the memory responsive to the communication traffic prior to the failure associated with the first link

218. To the extent "edge switch" in [16B] refers to the "network switching device" of claim 13, Gai's protocol entity 232 meets [16B] for the reasons explained above with respect to [1C] and [13C].

c. [16C] and, responsive to placing the second access port in the active state, to update the edge switch database so that entries in the edge switch database point to the second access port instead of the first access port.

219. To the extent "edge switch" in [16C] refers to the "network switching device" of claim 13, Gai's protocol entity 232 meets [16C] for the reasons explained above with respect to claim 4.

6. **Claim 17: A device according to claim 16, wherein the protocol processor is adapted to update the entries in the database to point to the second access port before receiving the communication traffic from the one or more access switches on the second port.**

220. Gai's protocol entity 232 meets claim 17 for the reasons explained above with respect to claim 5.

VI. CONCLUSION

I declare that all statements made herein of my own knowledge are true, that all statements made on information and belief are believed to be true, and that these statements were made with the knowledge that willful false statements and the like are punishable by fine or imprisonment, or both, under Section 1001 of Title 18 of the United States Code.

I declare under penalty of perjury that the foregoing is true and correct.

Dated: May 3, 2024



Bruce McNair